(Anonymous) Fully Secure Constant-Size Ciphertext HIBE From Type-3 Pairings

Somindu C. Ramanna Palash Sarkar

Applied Statistics Unit
Indian Statistical Institute
203, B.T. Road, Kolkata
India 700108.
e-mail: {somindu_r,palash}@isical.ac.in

Abstract

This paper takes a comprehensive look at hierarchical identity-based encryption (HIBE) schemes with constant-size ciphertexts which provide desirable provable guarantees and can be implemented using the most efficient bilinear pairings, namely, Type-3 pairings. Our work results in six new HIBE schemes. The first construction, *DPVS-HIBE*, is obtained by extending the IBE scheme of Chen *et.al.* The extension uses an identity-hashing technique introduced by Boneh, Boyen and Goh (BBG) in 2005. The second construction, *DPVS-AHIBE*, is obtained by appending the anonymity property to *DPVS-HIBE* via a method introduced by Boyen and Waters and later used in the context of constant-size ciphertext HIBE by Seo *et.al.* Scheme *DPVS-HIBE* does not allow the identity components to assume the value zero. It is extended to *DPVS-HIBE*-0 that allows zero to be an identity component. The construction *DPVS-AHIBE*-0 results from adding anonymity to *DPVS-HIBE*-0. The rest of the two constructions *LW-HIBE* and *LW-AHIBE* are obtained by extending the Type-3 pairing based IBE scheme due to Lewko and Waters (2010). All six schemes are proved secure against adaptive-identity attacks using the technique of dual system encryption and dual pairing vector spaces. Based on currently known techniques, these schemes make a major contribution to the state-of-the-art on efficient HIBE constructions.

Keywords: hierarchical identity-based encryption (HIBE), constant-size ciphertext, Type-3 pairings, dual pairing vector spaces, dual-system encryption.

1 Introduction

The notion of identity-based encryption (IBE) was introduced by Shamir [Sha84] and the first IBE schemes appeared later [Coc01, BF03]. In IBE, a sender encrypts a message using the receiver's identity itself as the public key and a central authority called private key generator (PKG) generates and securely distributes decryption keys corresponding to identities of different users. Hierarchical IBE (HIBE) was proposed by [GS02, HL02] to reduce the workload of the PKG. In a HIBE scheme, the PKG is allowed to delegate the key generation ability to lower-level entities. As a result, an individual user can conveniently obtain a decryption key from a lower-level entity instead of obtaining it from the PKG.

Type-3 Pairings: Most practical (H)IBE schemes are built using a bilinear pairing which maps $\mathbb{G}_1 \times \mathbb{G}_2$ to $\mathbb{G}_T$, where $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ are groups of the same order. The only known examples of such maps arise by suitably choosing $\mathbb{G}_1$ and $\mathbb{G}_2$ to be groups of elliptic curve points and $\mathbb{G}_T$ to be a subgroup of

the multiplicative group of a finite field. From an implementation point of view, it is most efficient to use bilinear maps where the (common) group order is prime and it is computationally difficult to find an isomorphism from $\mathbb{G}_1$ to $\mathbb{G}_2$ or vice versa. Such pairings are called Type-3 pairings [CM11, SV07, GPS08]. Less efficient alternatives are when $\mathbb{G}_1$ and $\mathbb{G}_2$ are same (called Type-1 pairings) or when the common group order is a composite number (called composite-order pairings). IBE or HIBE schemes based on Type-3 pairings would have the fastest algorithms and the most compact representations of parameters.

Constant-Size Ciphertext HIBE: In HIBE, identities consist of tuples of varying lengths. Encryption of a message is done for a specified identity tuple. In many HIBE schemes, as the length of the identity tuple increases, so does the length of the resulting ciphertext. Consequently, the bandwidth requirement for communicating the ciphertext also increases. For an individual user, this may actually lead to a disincentive for obtaining decryption keys from lower-level entities. The net result may be in opposition to the basic motivation for a HIBE which is the ability to off-load work to lower-level entities.

The solution to this issue is to require that the ciphertext size is independent of the length of the identity tuple. Then, irrespective of the length of the identity tuple, the bandwidth required for the ciphertext would be the same. Such a scheme is called a constant-size ciphertext HIBE. The first such HIBE scheme was proposed by Boneh, Boyen and Goh [BBG05]. While the scheme itself is quite elegant, its proof of security was in a very restricted attack model, the so-called selective-identity model. Lewko and Waters [LW10] provided the first constant-size ciphertext HIBE scheme which is secure against the usual adaptive-identity attacks. The drawback, however, was that the scheme in [LW10] used pairings on composite order groups and could not be instantiated with the more efficient Type-3 pairings.

In the following, we use the abbreviation CC-HIBE to denote HIBE schemes with constant-size ciphertexts. We clarify that the constant size here only refers to the number of groups elements in the ciphertext. The size of representation of the group elements, however, needs to increase if the value of the security parameter increases.

Anonymity: (H)IBE schemes with anonymity do not reveal any information about the identity of the recipient of a ciphertext. Abdalla *et.al.* [ABC$^+$05] provided the first formalisation of this notion and how this property enables construction of public key encryption and keyword search (PEKS) schemes. PEKS enables search to be performed on an encrypted document based on some keywords and this capability for search is delegateable. Anonymous HIBE schemes provide means to extend PEKS to more sophisticated primitives such as public key encryption with temporary keyword search (PETKS) and identity-based encryption with keyword search (IBEKS). The first construction of anonymous HIBE without random oracles was given by [BW06] with security in the selective-id model. Later construction by [SKOS09, DCIP10] could achieve security in the adaptive-id setting but were based on composite-order pairings. Another construction was given in [Duc10] and used asymmetric pairing but was again restricted to the selective-id model.

1.1 Our Contributions

Constant-size ciphertext HIBE is an important primitive. The goal of this paper is to comprehensively explore different possible constructions of this primitive using Type-3 pairings while guaranteeing nice provable properties, namely, security against adaptive-identity attacks, use of standard hardness assumptions, no degradation of security with increase in the depth of the HIBE and the avoidance of random oracles. To this end, we present six constant-size ciphertext HIBE schemes: *DPVS-HIBE*, *DPVS-AHIBE*, *LW-HIBE*, *LW-AHIBE*, *DPVS-HIBE*-0 and *DPVS-AHIBE*-0.

At a very basic level, the idea for obtaining constant-sized ciphertexts is to use the identity hashing technique suggested in [BBG05] over existing IBE schemes. We will refer to this as BBG-hash or BBG-extension. The method of obtaining anonymity is based on the ideas introduced in [BW06] and later used by [SKOS09] in the context of CC-HIBE. The nice provable properties have been established by combining the extremely useful idea of dual-system encryption [Wat09]. Four of the constructions are based on dual pairing vector spaces introduced in [OT08, OT09]. These are elegant mathematical structures that can be instantiated with Type-3 pairings having properties that are found in composite order pairings thus being suitable for constructions within the dual system framework. We next provide a brief description of how each of the six schemes is obtained.

$\mathcal{DPVS}$-$\mathcal{HIBE}$: Chen *et.al.* [CLL+12] constructed an IBE scheme based on dual pairing vector spaces using Type-3 pairings. IBE based on dual pairing vector spaces was earlier constructed by Lewko [Lew12] in the setting of Type-1 pairings. Till date, there has been no work that discusses the construction of CC-HIBE based on dual pairing vector spaces. We address this question by starting with the IBE of [CLL+12] and then extending it to a HIBE via BBG-extension. The main challenge is to define the public parameters and keys while keeping the dimension of the underlying dual pairing vector space constant since this determines the length of the ciphertext. Further, the last stage of the security proof is different from that in [CLL+12]. We provide a reduction based on a computational assumption whereas they provide an information theoretic argument. This is a result of their IBE scheme being anonymous and our HIBE extension being inherently non-anonymous.

$\mathcal{DPVS}$-$\mathcal{AHIBE}$: In order to make $\mathcal{DPVS}$-$\mathcal{HIBE}$ anonymous, we adapt the technique of [BW06] which was later used in [SKOS09, DCIP10]. The main idea is to provide enough information in the secret keys so that delegation can be done without using any elements from the public parameters. This requires providing two copies of the identity hash - one for decryption and one for re-randomisation during delegation. Though it appears straightforward to apply this technique, a problem arises while tackling the security argument. The problem is due to the fact that in each step of the hybrid argument (of the dual system framework) we can handle only two identities, the challenge identity and an identity provided for key extraction. Since a single secret key has two copies of the identity hash, each separately randomised, they have to be dealt with in separate games. We successfully get around this problem by using an alternate game sequence for the security proof.

$\mathcal{LW}$-$\mathcal{HIBE}$: In [LW10], Lewko and Waters constructed an IBE scheme based on composite-order pairings and extended it to a CC-HIBE scheme. They went on to convert their composite-order pairing based IBE scheme to one which can be instantiated using Type-3 prime-order pairings, which we will refer to as LW-IBE. However, no such conversion was done for the HIBE scheme in [LW10] and the authors do not make any remark on whether this can be done or how difficult it would be to do so. Instead of converting the composite-order HIBE, we start with LW-IBE and extend it to a CC-HIBE by plugging in the BBG-hash. One complication in doing so arises - extending the decryption key of LW-IBE to the decryption key of a HIBE in a straightforward manner does not work. In the dual-system technique, one has to appropriately define semi-functional components which have to be generated by the simulator while handling key extraction queries in the security reduction. Our way of tackling this is to add additional components to the decryption key. On the face of it, this complicates the key generation and delegation mechanisms. However, adding this extra level of complication allows the security reductions to go through.

LW-AHIBE: This scheme is obtained by extending *LW-HIBE* to achieve anonymity in manner similar to the way *DPVS-HIBE* is extended to construct *DPVS-AHIBE*. But the security proof is a bit different. The first and reductions proceed similar to that of *DPVS-AHIBE*. The third reduction is split into two stages - one based on DBDH and the other based on a new assumptions that we introduce called A1 (see Sections A.3 and D.5 for details).

DPVS-HIBE-0: All CC-HIBE schemes in the literature that use BBG-type identity-hash, as also the above four schemes, are not secure if the components of an identity tuple are allowed to take the value 0. If such schemes are improperly implemented so that an explicit check on identitites for 0 value is not carried out, then there is a simple and devastating attack as described in Section E. Forgetting to check for zero is a minor implementation error leading to a major security flaw. To emphasize that such omission is indeed possible, we would like to point out that even the theoretical papers [DCIP10, LW10] proposing CC-HIBE schemes do *not* explicitly mention that 0 should be disallowed as an identity component even though the schemes are insecure if identity components can take the value 0.

The problem of augmenting the BBG-HIBE scheme to allow for zero values in the identity components has been tackled in [CS07]. The idea was to use a different identity-hash while retaining the constant size of ciphertexts. We adapt this technique to convert *DPVS-HIBE* to a HIBE scheme named *DPVS-HIBE*-0 that allows 0 components in the identity-tuple. This, however, comes at a cost. The sizes of the public parameters and decryption keys get blown up by a factor of 2, though, there is no change in the efficiencies of encryption and decryption. Whether the increase in the sizes of the parameters is indeed tolerable will depend on the context. Our work reinforces the idea in [CS07] that 0 can indeed be allowed as an identity component and that there is an associated trade-off.

DPVS-AHIBE-0: This scheme is obtained by converting *DPVS-HIBE*-0 to accommodate anonymity property using the same path we followed to construct *DPVS-AHIBE* and *LW-AHIBE*. We believe that taken together, the above six constructions fill an important gap in the current literature on HIBE schemes.

Organisation of the paper: This paper contains descriptions and security proofs for six HIBE constructions. As a result there are a lot of details. We have chosen to present the flow of ideas with respect to the *DPVS-HIBE*. In Section 3, we describe how to move from the short IBE in [CLL+12] to a CC-HIBE and the ideas required in the security reduction. The extension to *DPVS-AHIBE* is given in Section 4 which also provides idea of how the security proof is modified. While working on this proof we have noted that the earlier application of the ideas in [BW06, SKOS09] to get an anonymous CC-HIBE using the dual system framework in [DCIP10] has a flaw in the proof. This flaw is described in the appendix in Section F.

The appendices describe the other four HIBE's, namely *LW-HIBE*, *LW-AHIBE*, *DPVS-HIBE*-0 and *DPVS-AHIBE*-0. The construction details and proofs of the *LW-HIBE* and the *LW-AHIBE* HIBE schemes are given in Section D of the appendix. Construction of *DPVS-HIBE*-0 is given in Section E.1 and the proof of security is provided in Section E.2. The anonymous version, *DPVS-AHIBE*-0, is described in Section E.3. The proof of *DPVS-AHIBE*-0 is obtained by modifying the proof of *DPVS-HIBE*-0 in exactly the same way that the proof of *DPVS-AHIBE* is obtained from the proof of *DPVS-HIBE*. Hence we omit the details.

1.2 Improvements Over Existing Schemes

Table 1 provides a comparison of *LW-HIBE*, *DPVS-HIBE* and *DPVS-HIBE*-0 with previously proposed non-anonymous CC-HIBE schemes. Table 2 compares *LW-AHIBE*, *DPVS-AHIBE* and *DPVS-AHIBE*-0 with

previous anonymous HIBE schemes.

Scheme	[BBG05]	[CS06]	[CS07]	[LW10]	*LW-HIBE*	*DPVS-HIBE*	*DPVS-HIBE*-0
Pairing	Type-1	Type-1	Type-1	Composite	Type-3	Type-3	Type-3
#pp	$(h+4,0)$	$(h+3+hk,0)$	$(2h+3,1)$	$(h+3,1)$	$(4h+11,1)$	$(8(h+1),1)$	$(16h,h)$
#msk	1	1	1	1	1	4	4
#cpr	2	2	3	2	6	4	4
#key	$h-\ell+2$	$(k+1)(h-\ell)+2$	$2(h-\ell+1)$	$h-\ell+2$	$3(h-\ell+2)$	$4(h-\ell+1)$	$8h-8\ell+4$
Enc	$(\ell+2,1)$	(2,1)	$(\ell+2,1)$	$(\ell+2,1)$	$(3(\ell+2),1)$	$(4(\ell+1),1)$	$(4(\ell+1),1)$
Dec	2	2	2	2	6	4	4
KGen	$h+2$	$2(h-\ell+1)$	$2h-\ell+2$	$2h-\ell+4$	$4h-3\ell+8$	$4(h+1)$	$8h-4\ell+4$
Deleg.	$\ell+2$	$2(h-\ell)$	$2h-\ell+1$	$2h-\ell+6$	$3h-2\ell+12$	$4(h+2)$	$2h-\ell+1$
Security	selective-id	adaptive-id	selective$^+$-id	adaptive-id	adaptive-id	adaptive-id	adaptive-id
Assump.	Decision h-wBDHI	h-wDBDHI*	h-wDBDHI*	Subgroup Decision	LW1,LW2, DBDH	XDH	XDH
Deg.	1	$O((kq2^{N/k})^h)$	1	$O(q)$	$O(q)$	$O(q)$	$O(q)$
Zero-ID	No	Yes	Yes	No	No	No	Yes

Table 1: Comparison of non-anonymous CC-HIBE schemes based on pairings without random oracles. h: maximum depth of the HIBE; ℓ: length of the identity tuple; q: no. of key extraction queries; N ([CS06]): number of bits in an identity; k ([CS06]): number of blocks of N/k bits; #pp, #msk, #cpr, #key - number of group elements in the public parameters, master secret, ciphertext and key respectively. Enc, Dec, KGen, Deleg - efficiency of encryption, decryption, key generation and delegation algorithms. Type-3 pairing based schemes - $\mathcal{PP}$ and ciphertexts consist elements of $\mathbb{G}_1$; $\mathcal{MSK}$ and keys consist elements of $\mathbb{G}_2$. #pp $= (a, b)$ means that there are a elements of $\mathbb{G}_1$ and b elements of $\mathbb{G}_T$; Enc $= (a, b)$: a scalar multiplications in $\mathbb{G}_1$ and b exponentiations in $\mathbb{G}_T$; Dec: #pairings; KGen: #scalar multiplications in $\mathbb{G}_2$; Deleg: #scalar multiplications in $\mathbb{G}_2$. Assump: underlying complexity assumptions; Deg: security degardation; Zero-ID: whether the scheme allows zero to be an identity component or not.

Scheme	[BW06]	[SKOS09]	[DCIP10]	*LW-AHIBE*	*DPVS-AHIBE*	*DPVS-AHIBE*-0
Pairing	Type-1	Composite	Composite	Type-3	Type-3	Type-3
#pp	$(2(h^2+3h+2),1)$	$(h+6,1)$	$(h+4,1)$	$(3h+6,1)$	$(4(h+1),1)$	$(8h,h)$
#msk	h^2+5h+7	$h+4$	2	$h+6$	$4(h+2)$	$4(2h+1)$
#cpr	$2h+5$	3	2	6	4	4
#key	$(h+3)(3h-\ell+5)$	$3(h-\ell+3)$	$2(h-\ell+2)$	$6(h-\ell+2)$	$8(h-\ell+2)$	$8(2h-2\ell+3)$
Enc	$(2(\ell+3)(h+2)+1,1)$	$(\ell+6,1)$	$(\ell+4,1)$	$(3(\ell+2),1)$	$(4(\ell+1),1)$	$(4(\ell+1),1)$
Dec	$2h+3$	4	2	6	4	4
KGen	$h^3+h^2(5-\ell)+$ $h(7-3\ell)-2\ell+2$	$3h-2\ell+2$	$4(h+2-3\ell)$	$6h-5\ell+12$	$8h-7\ell+8$	$4(2h-\ell+2)$
Deleg.	$5(h+2)(h+3)+1$	$6(h-\ell)+21$	$4(h-\ell)+11$	$2(h-\ell+3)$	$8(h-\ell+1)$	$8(h-\ell+2)$
Security	selective-id	selective-id	adaptive-id	adaptive-id	adaptive-id	adaptive-id
Assump.	DLin,DBDH	ℓ-wBDH*, ℓ-cDH	Subgroup Decision	LW1,LW2, DBDH	XDH	XDH
Deg.	$O(1)$	$O(1)$	$O(q)$	$O(q)$	$O(q)$	$O(q)$
Zero-ID	No	No	No	No	No	Yes

Table 2: Comparison of anonymous HIBE schemes based on pairings without random oracles.

From the tables, one can see that the *LW-HIBE* is the most efficient among the non-anonymous HIBEs with all the provable properties except that the security is based on non-standard assumptions. The scheme *DPVS-HIBE* is less efficient than *LW-HIBE* but is secure under the standard XDH assumptions. Among the anonymous HIBE schemes, *LW-AHIBE* is the most efficient with security under non-standard assumptions and *DPVS-AHIBE* is the most efficient achieving all the provable guarantees. The main advantages of the dual system framework is that security can be based on static assumptions and the degradation is

independent of the depth of the HIBE.

In absolute terms, the number of group elements required for composite-order based schemes is less than that required in the our HIBE schemes. However, only counting group elements is not a proper comparison. One has to consider the actual size for representing a single group element at a desired security level.

For concreteness, let us consider a security level of 128 bits. For Type-3 pairings, using Table-2 of [CHKM10], elements of $\mathbb{G}_1$ and $\mathbb{G}_2$ can be represented using 257 and 513 bits respectively. In contrast, the order of $\mathbb{G}_1 = \mathbb{G}_2$ for composite-order pairings is a product of at least three primes. The basic security requirement is that this group order should be hard to factor. To attain 128-bit security level, the length of the bit representation of the group order should be about 3000 bits (or more). So, for schemes based on composite-order groups, the length of representations of elements of $\mathbb{G}_1$ (and $\mathbb{G}_2$) will be about 3000 bits. This is about 12 times (resp. 6 times) more than the length of bit representation of elements of $\mathbb{G}_1$ (resp. $\mathbb{G}_2$) using Type-3 pairings. The wide difference in the length of representations of group elements more than adequately compensates for the absolute number of group elements in composite-order HIBE schemes being lesser than that in the newly proposed HIBE scheme.

For example, ciphertexts in $\mathcal{LW}$-$\mathcal{HIBE}$ consist of 6 elements of $\mathbb{G}_1$ which is about 1500 bits whereas ciphertexts in the HIBE of [LW10] will be about 9000 bits (3 elements each having length about 3000 bits). Similar considerations apply to public parameters ($\mathcal{PP}$), master secret key ($\mathcal{MSK}$) and decryption keys.

2 Preliminaries

Here we mention a few notation and definitions. The rest including the complexity assumptions used in the proofs are given in Section A of the appendix.

2.1 Notation

For a set $\mathcal{X}$, the notation $x_1, \ldots, x_k \in_{\mathrm{R}} \mathcal{X}$ indicates that $x_1, \ldots, x_k$ are independent random elements of $\mathcal{X}$ with some distribution R and $x_1, \ldots, x_k \xleftarrow{\mathrm{R}} \mathcal{X}$ symbolises $x_1, \ldots, x_k$ being sampled independently from $\mathcal{X}$ according to R. The uniform distribution is denoted by U. For a (probabilistic) algorithm $\mathcal{A}$, $x \longleftarrow \mathcal{A}(\cdot)$ means that x is chosen according to the output distribution of $\mathcal{A}$ (which of course may be determined by its input). For a prime p, we denote by $\mathbb{Z}_p$ the field of order p, $\mathbb{Z}_p^\times = \mathbb{Z}_p \setminus \{0\}$ and $\mathbb{Z}_p^n$ the vector space of all n-tuples over $\mathbb{Z}_p$. Vectors over $\mathbb{Z}_p$ will be represented by bold face lower case letters (e.g. $\mathbf{v}$). For vectors $\mathbf{u} = (u_1, \ldots, u_n)$ and $\mathbf{v} = (v_1, \ldots, v_n)$, $\mathbf{u} \cdot \mathbf{v}$ denotes their inner product $\sum_{i=1}^n u_i v_i$. $\mathbb{Z}_p^{n \times n}$ denotes the set of all $n \times n$ matrices over $\mathbb{Z}_p$ and $GL(n, \mathbb{Z}_p)$, the general linear group of degree n over $\mathbb{Z}_p$. Let $\mathbb{V}$ and $\mathbb{V}^*$ be vector spaces over $\mathbb{Z}_p$ obtained by direct product of groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of order p. Vectors in $\mathbb{V}$ shall be expressed using arrow symbols over capitalised letters, for example $\vec{P}$. The asterisk symbol is used to differentiate between vectors of $\mathbb{V}$ and $\mathbb{V}^*$ with the same names (e.g. $\vec{P}$ and $\vec{P}^*$). We denote matrices over $\mathbb{Z}_p$ by bold face upper case letters (e.g. $\mathbf{A}$) and matrices over $\mathbb{G}_1$ (resp. $\mathbb{G}_2$) by blackboard bold font, e.g. $\mathbb{A}$ (resp. $\mathbb{A}^*$). $\mathbf{A}^T$ denotes the transpose of matrix $\mathbf{A}$.

2.2 Definitions

Definition 2.1. Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be cyclic groups of prime order p. $\mathbb{G}_1$,$\mathbb{G}_2$ are additive groups and $\mathbb{G}_T$ is multiplicative. A pairing is a tuple $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, P_1, P_2)$ where $\mathbb{G}_1 = \langle P_1 \rangle$, $\mathbb{G}_2 = \langle P_2 \rangle$ and

$e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is an efficiently computable non-degenerate bilinear map i.e., for elements $Q_1 \in \mathbb{G}_1$ and $Q_2 \in \mathbb{G}_2$, $e(Q_1, Q_2) = 1$ iff $Q_1 = 0$ or $Q_2 = 0$ and for $x, y \in \mathbb{Z}_p$, $e(xP_1, yP_2) = e(P_1, P_2)^{xy}$.

A bilinear map is called symmetric or a Type-1 bilinear map if $\mathbb{G}_1 = \mathbb{G}_2$; otherwise it is asymmetric. Asymmetric bilinear maps are further classified into Type-2 and Type-3 bilinear maps. In the Type-2 setting, there is an efficiently computable isomorphism either from $\mathbb{G}_1$ to $\mathbb{G}_2$ or from $\mathbb{G}_2$ to $\mathbb{G}_1$ whereas in the Type-3 setting there are no such isomorphisms known. We will denote elements of $\mathbb{G}_1$ by upper case letters with subscript 1 and elements of $\mathbb{G}_2$ with subscript 2.

A formal definition of dual pairing vector spaces is provided in Section A.1. Here we require only an instantiation of DPVS using asymmetric pairings.

Dual Pairing Vector Space from Asymmetric Pairings. A typical construction for a DPVS $\mathcal{V}_n = (p, \mathbb{V}, \mathbb{V}^*, \mathbb{G}_T, \bar{e}, \mathbb{A}, \mathbb{A}^*)$ is via a direct product over a pairing $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, P_1, P_2)$. We consider such constructions over asymmetric pairings, in particular, Type-3 pairings because they have more efficient implementations both in terms of representation of group elements and pairing computation. Details of the construction are provided below.

- $\mathbb{V} = \underbrace{\mathbb{G}_1 \times \cdots \times \mathbb{G}_1}_{n \text{ times}}$ and $\mathbb{V}^* = \underbrace{\mathbb{G}_2 \times \cdots \times \mathbb{G}_2}_{n \text{ times}}$ are vector spaces of dimension n over $\mathbb{Z}_p$. Their vectors can be expressed as $\vec{X} = \mathbf{x}P_1 = (x_1P_1, \ldots, x_nP_1)$ and $\vec{Y} = \mathbf{y}P_2 = (y_1P_2, \ldots, y_nP_2)$ respectively, where $\mathbf{x}, \mathbf{y} \in \mathbb{Z}_p^n$.
- $\mathbb{A}, \mathbb{A}^*$ form the canonical bases of $\mathbb{V}, \mathbb{V}^*$ respectively i.e., $\mathbb{A} = (\vec{A}_1, \ldots, \vec{A}_n)$ and $\mathbb{A}^* = (\vec{A}_1^*, \ldots, \vec{A}_n^*)$ with $\vec{A}_i = (\underbrace{0, \ldots, 0}_{i-1}, P_1, \underbrace{0, \ldots, 0}_{n-i})^T$ and $\vec{A}_i^* = (\underbrace{0, \ldots, 0}_{i-1}, P_2, \underbrace{0, \ldots, 0}_{n-i})^T$ for $i = 1, \ldots, n$.
- Let $\vec{X} = \mathbf{x}P_1$ and $\vec{Y} = \mathbf{y}P_2$. The function $\bar{e}$ is defined as

$$\bar{e}(\vec{X}, \vec{Y}) = \prod_{i=1}^{n} e(x_iP_1, y_iP_2) = \prod_{i=1}^{n} e(P_1, P_2)^{x_iy_i} = e(P_1, P_2)^{\mathbf{x} \cdot \mathbf{y}},$$

  Note that $\bar{e}(\mathbf{x}P_1, \mathbf{y}P_2) = 1$ iff $\mathbf{x} \cdot \mathbf{y} = 0$.

Dual bases. For a fixed dimension n, let $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_n)$ and $\mathbf{B}^* = (\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*)$ be two bases of $\mathbb{Z}_p^n$ such that $\mathbf{b}_i \cdot \mathbf{b}_j^* = 0$ for $i \neq j$ and $\mathbf{b}_i \cdot \mathbf{b}_i^* = \psi$ for all $i \in \{1, 2, \ldots, n\}$ where ψ is an element of $\mathbb{Z}_p^\times$. Then $(\mathbf{B}, \mathbf{B}^*)$ are called dual bases of $\mathbb{Z}_p^n$. It can be shown that for a given basis $\mathbf{B}$, its dual basis is determined by $\mathbf{B}^* = (\mathbf{B}^{-1})^T$.

There is an efficient algorithm $\mathsf{Dual}(\mathbb{Z}_p^n)$ that returns random dual bases $(\mathbf{B}, \mathbf{B}^*)$ of $\mathbb{Z}_p^n$. Let $(\mathbf{A}, \mathbf{A}^*)$ denote the canonical dual bases of $\mathbb{Z}_p^n$ i.e., $\mathbf{A} = (\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n)$ with $\mathbf{a}_i = (\underbrace{0, \ldots, 0}_{i-1}, 1, \underbrace{0, \ldots, 0}_{n-i})$ and $\mathbf{A}^* = (\mathbf{A}^{-1})^T$. The algorithm Dual chooses a matrix $\mathbf{X} \xleftarrow{\mathrm{U}} GL(n, \mathbb{Z}_p)$ and computes a set of dual bases as follows: $\mathbf{B} = \mathbf{A}\mathbf{X}$ and $\mathbf{B}^* = \mathbf{A}^*(\mathbf{X}^T)^{-1}$. It essentially applies a linear transformation to $\mathbf{A}$ and the corresponding changes to $\mathbf{A}^*$ to retain the duality property.

3 CC-HIBE from Dual Pairing Vector Spaces

3.1 Short IBE of Chen *et.al.*

A description of the IBE scheme proposed by Chen *et.al.* [CLL+12] is provided here. We use a compact notation to denote normal and semi-functional ciphertexts and keys. The group elements shown in curly brackets {} are the semi-functional components. To get the scheme itself, these components should be ignored.

The description is in terms of a Type-3 pairing $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, P_1, P_2)$ and random dual bases $(\mathbf{B}, \mathbf{B}^*)$ with $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3, \mathbf{b}_4)$ and $\mathbf{B}^* = (\mathbf{b}_1^*, \mathbf{b}_2^*, \mathbf{b}_3^*, \mathbf{b}_4^*)$ where each vector is an element of $\mathbb{Z}_p^4$. The public parameters, the master secret key, the decryption key and the ciphertext are as follows. α, s and r are uniform random elements of $\mathbb{Z}_p$.

$$
\begin{aligned}
&\mathcal{PP} \quad : (\mathbf{b}_1 P_1, \mathbf{b}_2 P_1, e(P_1, P_2)^{\alpha \mathbf{b}_1 \cdot \mathbf{b}_1^*}).\\
&\mathcal{MSK}\text{: } (\alpha, \mathbf{b}_1^* P_2, \mathbf{b}_2^* P_2).\\
&\mathcal{C} = \Big(C_0 = M \cdot e(P_1, P_2)^{s\alpha \mathbf{b}_1 \cdot \mathbf{b}_1^*},\ \vec{C_1} = s\mathbf{b}_1 P_1 + s\mathsf{id}\mathbf{b}_2 P_1 \{ + t_1 \mathbf{b}_3 P_1 + t_2 \mathbf{b}_4 P_1 \} \Big).\\
&\mathcal{SK}_{\mathsf{id}} = \alpha \mathbf{b}_1^* P_2 + r\mathsf{id}\mathbf{b}_1^* P_2 - r\mathbf{b}_2^* P_2 \{ + z_1 \mathbf{b}_3^* P_2 + z_2 \mathbf{b}_4^* P_2 \}.
\end{aligned}
$$

The randomisers for the semi-functional components are t_1, t_2 and z_1, z_2 in the ciphertext and key respectively.

3.2 Moving to Constant-Size Ciphertext HIBE

The starting point for this construction is the 'short' IBE of [CLL+12] which is described in Section 3.1. It is based on dual pairing vector spaces over asymmetric pairing groups in which the XDH assumption holds. As earlier, identity for an entity at depth ℓ is a vector in $(\mathbb{Z}_p^\times)^\ell$. So, the role of id in the short IBE is to be played by some hash of the identity vector $\mathbf{id}$. The identity hash we use is the one previously used by Boneh, Boyen and Goh [BBG05].

In BBG-HIBE, there are $h+1$ group elements provided in the public parameters that are used to hash the identity vector. There are two ways to adapt this in the setting of dual pairing vector spaces. One method is to work with a vector space $\mathbb{V}$ of dimension at least $h+1$ and provide $h+1$ elements of a basis of $\mathbb{V}$. But this leads to the ciphertext to have vectors (although constant in number) with length at least $h+1$ which is linear in the maximum depth of the HIBE. Also this leads to number of group elements in the public parameters to be $O(h^2)$. So, in order to keep the number of group elements in the ciphertext constant, we considered constant-dimension (four) vector spaces as in [CLL+12] and to hash the identity, we provide h vectors $\theta_1 \mathbf{b}_2 P_1, \ldots, \theta_h \mathbf{b}_2 P_1$ from the span of $\mathbf{b}_2$ along with $\mathbf{b}_1$ as part of the public parameters. Here the θ_j's are chosen uniformly and independently at random. To encrypt to an ℓ-level identity only the first ℓ of these parameters will be required. Now, the role of id will be played by $\sum_{i=1}^{\ell} \theta_j \mathsf{id}_j$ for an ℓ-length identity vector $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$.

Having determined the public parameters, we consider how the decryption key is to be augmented. In the above description, the decryption key also has id which has to be replaced by $\sum_{i=1}^{\ell} \theta_j \mathsf{id}_j$. The PKG can keep $\theta_1, \ldots, \theta_h$ as part of the master secret key and compute the identity hash as and when required. But, then there will be a problem with delegation. Whenever an identity of depth ℓ wishes to create a decryption key for an identity of depth $\ell+1$, it has to perform a re-randomisation. To fulfil the requirement of re-randomisation two things have to be available – first, the vectors $\theta_1 \mathbf{b}_1^* P_2, \ldots, \theta_h \mathbf{b}_1^* P_2$

and second, the vectors $r\theta_{\ell+1}\mathbf{b}_1^*P_2, \ldots, r\theta_h\mathbf{b}_1^*P_2$ where r is the randomiser used to generate the decryption key for the level-ℓ identity. We write the vectors $\theta_1\mathbf{b}_1^*P_2, \ldots, \theta_h\mathbf{b}_1^*P_2$ as part of the master secret key, but, these have to be made available publicly for delegation. With the above modifications to the short IBE, we obtain the CC-HIBE described in the following section.

3.3 Construction of *DPVS-HIBE*

Setup (λ): Let h denote the maximum depth of the HIBE. Construct a DPVS $\mathcal{V}_4 = (p, \mathbb{V}, \mathbb{V}^*, \mathbb{G}_T, \bar{e}, \mathbb{A}, \mathbb{A}^*)$ from a Type-3 pairing $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, P_1, P_2)$ whose parameters are generated according to λ. Choose random dual bases $(\mathbf{B}, \mathbf{B}^*) \longleftarrow \mathsf{Dual}(\mathbb{Z}_p^4)$ and $\alpha, \theta_1, \theta_2, \ldots, \theta_h \xleftarrow{\mathrm{U}} \mathbb{Z}_p$. Set the public parameters and master secret as follows.

$$\mathcal{PP} \quad : (\mathbf{b}_1P_1,\ \theta_1\mathbf{b}_2P_1,\ \ldots,\ \theta_h\mathbf{b}_2P_1,\ e(P_1,P_2)^{\alpha\mathbf{b}_1\cdot\mathbf{b}_1^*}).$$
$$\mathcal{MSK}\text{: } (\alpha\mathbf{b}_1^*P_2,\ \theta_1\mathbf{b}_1^*P_2, \ldots,\ \theta_h\mathbf{b}_1^*P_2,\ \mathbf{b}_2^*P_2).$$

The only quantity of the $\mathcal{MSK}$ which is required to be kept secret by the PKG is $\alpha\mathbf{b}_1^*P_2$. The remaining components of the $\mathcal{MSK}$ are not kept secret. The remaining components of the $\mathcal{MSK}$ are not kept secret. The vectors $(\theta_1\mathbf{b}_1^*P_2, \ldots,\ \theta_h\mathbf{b}_1^*P_2,\ \mathbf{b}_2^*P_2)$ are indeed required for key delegation and can be maintained by the PKG on its public database. During encryption, only $\mathcal{PP}$ would be downloaded, whereas for key delegation the vectors $(\theta_1\mathbf{b}_1^*P_2, \ldots,\ \theta_h\mathbf{b}_1^*P_2,\ \mathbf{b}_2^*P_2)$ would be downloaded.

$\mathsf{Encrypt}$ $(\mathcal{PP}, M, \mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell))$: Choose a randomiser s from $\mathbb{Z}_p$. The ciphertext consists of an element of C_0 of $\mathbb{G}_T$ and a vector $\vec{C}_1$ from $\mathbb{V}$ as follows.

$$C_0 = M \cdot e(P_1,P_2)^{s\alpha\mathbf{b}_1\cdot\mathbf{b}_1^*}, \quad \vec{C}_1 = s\mathbf{b}_1P_1 + s\textstyle\sum_{j=1}^{\ell}(\mathsf{id}_j\theta_j\mathbf{b}_2P_1).$$

KeyGen $(\mathcal{PP}, \mathcal{MSK}, \mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell))$: Choose $r \xleftarrow{\mathrm{U}} \mathbb{Z}_p$. The secret key for $\mathbf{id}$, $\mathcal{SK}_{\mathbf{id}}$ is given by $\mathcal{SK}_{\mathbf{id}} = (\vec{K}, \vec{D}_{\ell+1}, \ldots, \vec{D}_h)$ where

$$\vec{K} = \alpha\mathbf{b}_1^*P_2 + r\textstyle\sum_{j=1}^{\ell}(\mathsf{id}_j\theta_j\mathbf{b}_1^*P_2) - r\mathbf{b}_2^*P_2,\ \vec{D}_{\ell+1} = r\theta_{\ell+1}\mathbf{b}_1^*P_2, \quad \ldots \quad , \vec{D}_h = r\theta_h\mathbf{b}_1^*P_2.$$

The vector K is used for actual decryption and the remaining vectors are used for delegation.

$\mathsf{Delegate}$ $(\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{SK}_{\mathbf{id}}, \mathsf{id}_{\ell+1}, \mathcal{PP})$: Define the notation $\mathbf{id} : \mathsf{id}_{\ell+1} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell, \mathsf{id}_{\ell+1})$, i.e., $\mathbf{id} : \mathsf{id}_{\ell+1}$ denotes the tuple obtained by concatenating $\mathsf{id}_{\ell+1}$ to the end of $\mathbf{id}$.

Pick $\tilde{r} \xleftarrow{\mathrm{U}} \mathbb{Z}_p$. The secret key for the identity $\mathbf{id} : \mathsf{id}_{\ell+1}$ is computed as follows:

$$\vec{K} \leftarrow \vec{K} + \mathsf{id}_{\ell+1}\vec{D}_{\ell+1} + \tilde{r}\textstyle\sum_{j=1}^{\ell+1}(\mathsf{id}_j\theta_j\mathbf{b}_1^*P_2) - \tilde{r}\mathbf{b}_2^*P_2,\ \vec{D}_j \leftarrow \vec{D}_j + \tilde{r}\theta_j\mathbf{b}_1^*P_2 \text{ for } \ell+2 \le j \le h.$$

thus setting the randomiser to $r + \tilde{r}$ for the new secret key given by $\mathcal{SK}_{\mathbf{id}:\mathsf{id}_{\ell+1}} = (\vec{K}, \vec{D}_{\ell+2}, \ldots, \vec{D}_h)$.

$\mathsf{Decrypt}$ $(\mathcal{C}, \mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{SK}_{\mathbf{id}}, \mathcal{PP})$: Decryption is done as follows: $M = C_0/\bar{e}(\vec{C}_1, \vec{K})$.

Correctness of decryption. The following computation shows the correctness of decryption.

$$\begin{aligned}
\bar{e}(\vec{C}_1, \vec{K}) &= \bar{e}\left(s\mathbf{b}_1 P_1 + s\sum_{j=1}^{\ell}(\mathsf{id}_j\theta_j\mathbf{b}_2 P_1),\ \alpha\mathbf{b}_1^* P_2 + r\sum_{j=1}^{\ell}(\mathsf{id}_j\theta_j\mathbf{b}_1^* P_2) - r\mathbf{b}_2^* P_2\right) \\
&= \bar{e}\left(\mathbf{b}_1 P_1,\ \mathbf{b}_1^* P_2\right)^{s\alpha} \cdot \bar{e}\left(\mathbf{b}_1 P_1,\ \sum_{j=1}^{\ell}(\mathsf{id}_j\theta_j\mathbf{b}_1^* P_2)\right)^{rs} \cdot \bar{e}\left(\sum_{j=1}^{\ell}(\mathsf{id}_j\theta_j\mathbf{b}_2 P_1,\ \mathbf{b}_2^* P_2\right)^{-rs} \\
&= e\left(P_1,\ P_2\right)^{s\alpha\mathbf{b}_1\cdot\mathbf{b}_1^*} \cdot e\left(P_1,\ P_2\right)^{rs(\mathbf{b}_1\cdot\mathbf{b}_1^*)\sum_{j=1}^{\ell}(\mathsf{id}_j\theta_j)} \cdot e\left(P_1,\ P_2\right)^{-rs(\mathbf{b}_2\cdot\mathbf{b}_2^*)\sum_{j=1}^{\ell}(\mathsf{id}_j\theta_j)} \\
&= e\left(P_1,\ P_2\right)^{s\alpha\mathbf{b}_1\cdot\mathbf{b}_1^*}
\end{aligned}$$

The second equality follows from the property of dual bases that $\mathbf{b}_i \cdot \mathbf{b}_j^* = 0$ for $i \neq j$ i.e., the cross-product terms $\mathbf{b}_1 \cdot \mathbf{b}_2^*$ and $\mathbf{b}_2 \cdot \mathbf{b}_1^*$ cancel out. The last equality follows from the fact that $\mathbf{b}_1 \cdot \mathbf{b}_1^* = \mathbf{b}_2 \cdot \mathbf{b}_2^* = \psi$.

3.4 Security

Following Waters' approach, an analysis following the dual-system technique has three reductions. Below we describe some intuition about the proof highlighting the similarities and differences of the security analysis of *DPVS-HIBE* with that of the short IBE scheme in [CLL+12].

Semi-functional components: The definition of semi-functional keys for the HIBE scheme is different. The vector $\vec{K}$ is defined the same way as in [CLL+12] where a random vector from $\mathsf{span}\langle\mathbf{b}_3^* P_2, \mathbf{b}_4^* P_2\rangle$ is added. Additionally, it is required to define semi-functional components for the vectors $\vec{D}_{\ell+1}, \ldots, \vec{D}_h$. This is because all these vectors share the same randomiser, r, with $\vec{K}$ and the randomiser comes from the assumption. For each of these vectors we add an independent random vector from $\mathsf{span}\langle\mathbf{b}_3^* P_2, \mathbf{b}_4^* P_2\rangle$ as the semi-functional component. The semi-functional ciphertexts are defined in exactly the same way as in [CLL+12].

First reduction: In the first reduction, the challenge ciphertext is replaced by a semi-functional ciphertext. The challenge ciphertext is computed using instance of subspace assumption DS1 (defined in A.3) so that it is hard to distinguish between an element of $\mathsf{span}\langle\mathbf{b}_1 P_1, \mathbf{b}_2 P_1\rangle$ and a random vector of $\mathbb{V}$ and hence a normal ciphertext from semi-functional.

Second reduction: In this reduction, it is shown that the adversary cannot decide whether the response to the k-th key extraction query is normal or semi-functional. The k-th key is constructed from an instance of DS2 assumption in such a way that when the instance is 'real' the vectors $\vec{K}, \vec{D}_{\ell+1}, \ldots, \vec{D}_h$ are properly distributed elements of the key and when the instance is 'random' these are random vectors of $\mathbb{V}^*$. In the case of HIBE, the key has additional components which are required for delegation and these have semi-functional components. The IBE scheme in [CLL+12] did not have these components. We show how to simulate these additional components of the decryption key.

Third reduction: Here, the message that the challenge ciphertext encrypts is changed to random element of $\mathbb{G}_T$. The short IBE scheme of [CLL+12] is anonymous and the third reduction is based on an information theoretic argument. In the case of *DPVS-HIBE*, however, the situation is different because it is not anonymous (see Section 4). So, the third reduction for the HIBE is different. We show that solving the problem DS1 is no harder than distinguishing between a semi-functional encryption of the real message from a semi-functional encryption of a random element of $\mathbb{G}_T$.

The detailed security proof is provided in Section B.

4 Achieving Anonymity

Consider the scheme *DPVS-HIBE*. In the anonymity game the adversary provides a message M and two identities $\mathbf{id}_1$,$\mathbf{id}_2$ and obtains an encryption of M under one of the identities $\mathbf{id}_\beta$ from the challenger. Let the ciphertext be $(C_0, \vec{C}_1)$. In order check whether $\beta = 0$, the adversary simply computes the quantity $\vec{U} = \sum_{j=1}^{\ell}(\mathsf{id}_j\theta_j\mathbf{b}_1^*P_2) - \mathbf{b}_2^*P_2$, where $\mathbf{id}_0 = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$, and then checks whether $\bar{e}(\vec{U}, \vec{C}_1) = C_0/M$. If the test fails then $\beta = 1$ and otherwise $\beta = 0$ thus giving the adversary maximum advantage in winning the anonymity game. Therefore, *DPVS-HIBE* is not anonymous. But it can be modified to achieve anonymity retaining the constant-size ciphertext property. There is, however, a trade-off. The cost of achieving anonymity is an increase in the size of the decryption key and a corresponding increase in time required for key generation and delegation. The ciphertext remains unchanged, a few elements from the public parameters are included in the master secret key.

The method of achieving anonymity is simple and is based on the construction in [BW06]. Consider the scheme *DPVS-HIBE* for instance. The vectors $\theta_1\mathbf{b}_1^*P_2, \ldots, \theta_h\mathbf{b}_1^*P_2$ are not actually part of the master secret key as they have to be made available to enable re-randomisation during delegation. On the other hand, it is due to making these elements publicly available that results in loss of anonymity. So, there is an apparent contradiction present here. The way out is to keep these elements secret as part of the master secret and provide suitably randomised copies of elements already present in the key to facilitate delegation. This means that the decryption key size now increases due to these extra vectors. With the availability of these extra vectors, it becomes possible to perform re-randomisation during key delegation.

Following section provides a description of the scheme *DPVS-AHIBE* obtained by extending *DPVS-HIBE* for realising anonymity.

4.1 Construction of *DPVS-AHIBE*

$\mathsf{Setup}\ (\lambda)$: Let h denote the maximum depth of the HIBE. Construct a DPVS $\mathcal{V}_4 = (p, \mathbb{V}, \mathbb{V}^*, \mathbb{G}_T, \bar{e}, \mathbb{A}, \mathbb{A}^*)$ from a Type-3 pairing $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, P_1, P_2)$ whose parameters are generated according to λ. Choose random dual bases $(\mathbf{B}, \mathbf{B}^*) \longleftarrow \mathsf{Dual}(\mathbb{Z}_p^4)$ and $\alpha, \theta_1, \theta_2, \cdots, \theta_h \xleftarrow{\mathrm{U}} \mathbb{Z}_p$. Set the public parameters and master secret as follows.

$$\mathcal{PP} \ : (\mathbf{b}_1P_1,\ \theta_1\mathbf{b}_2P_1,\ \ldots,\ \theta_h\mathbf{b}_2P_1,\ e(P_1,P_2)^{\alpha\mathbf{b}_1\cdot\mathbf{b}_1^*}).$$
$$\mathcal{MSK}\text{: } (\alpha\mathbf{b}_1^*P_2,\ \theta_1\mathbf{b}_1^*P_2, \ldots,\ \theta_h\mathbf{b}_1^*P_2,\ \mathbf{b}_2^*P_2).$$

$\mathsf{Encrypt}\ (\mathcal{PP}, M, \mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell))$: Choose a randomiser s from $\mathbb{Z}_p$. The ciphertext consists of an element C_0 of $\mathbb{G}_T$ and a vector $\vec{C}_1$ from $\mathbb{V}$ as follows.

$$C_0 = M \cdot e(P_1, P_2)^{s\alpha\mathbf{b}_1\cdot\mathbf{b}_1^*}, \quad \vec{C}_1 = s\mathbf{b}_1P_1 + s\textstyle\sum_{j=1}^{\ell}(\mathsf{id}_j\theta_j\mathbf{b}_2P_1).$$

$\mathsf{KeyGen}\ (\mathcal{PP}, \mathcal{MSK}, \mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell))$: Choose $r_1, r_2 \xleftarrow{\mathrm{U}} \mathbb{Z}_p$. The secret key for $\mathbf{id}$, $\mathcal{SK}_{\mathbf{id}}$ is given by $\mathcal{SK}_{\mathbf{id}} = (\vec{K}_1, \vec{D}_{1,\ell+1}, \ldots, \vec{D}_{1,h}, \vec{K}_2, \vec{D}_{2,\ell+1}, \ldots, \vec{D}_{2,h})$ where

$$\vec{K}_1 = \alpha\mathbf{b}_1^*P_2 + r_1\textstyle\sum_{j=1}^{\ell}(\mathsf{id}_j\theta_j\mathbf{b}_1^*P_2) - r_1\mathbf{b}_2^*P_2,\ \vec{D}_{1,\ell+1} = r\theta_{\ell+1}\mathbf{b}_1^*P_2, \quad \ldots \quad, \vec{D}_{1,h} = r\theta_h\mathbf{b}_1^*P_2.$$
$$\vec{K}_2 = r_2\textstyle\sum_{j=1}^{\ell}(\mathsf{id}_j\theta_j\mathbf{b}_1^*P_2) - r_2\mathbf{b}_2^*P_2,\ \vec{D}_{2,\ell+1} = r_2\theta_{\ell+1}\mathbf{b}_1^*P_2, \quad \ldots \quad, \vec{D}_{2,h} = r_2\theta_h\mathbf{b}_1^*P_2.$$

The vector K_1 is used for decryption and the remaining vectors are used for delegation.

$\mathsf{Delegate}$ $(\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{SK}_{\mathbf{id}}, \mathsf{id}_{\ell+1}, \mathcal{PP})$: Define the notation $\mathbf{id} : \mathsf{id}_{\ell+1} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell, \mathsf{id}_{\ell+1})$, i.e., $\mathbf{id} : \mathsf{id}_{\ell+1}$ denotes the tuple obtained by concatenating $\mathsf{id}_{\ell+1}$ to the end of $\mathbf{id}$.

Pick $\tilde{r_1}, \tilde{r_2} \xleftarrow{\mathrm{U}} \mathbb{Z}_p$. The secret key for the identity $\mathbf{id} : \mathsf{id}_{\ell+1}$ is computed as follows:

$$\vec{K_1} \leftarrow \vec{K_1} + \mathsf{id}_{\ell+1}\vec{D}_{1,\ell+1} + \tilde{r_1}(\vec{K_2} + \mathsf{id}_{\ell+1}\vec{D}_{2,\ell+1}),\ \vec{D}_{1,j} \leftarrow \vec{D}_{1,j} + \tilde{r_1}\vec{D}_{2,j} \text{ for } \ell+2 \leq j \leq h.$$
$$\vec{K_2} \leftarrow \tilde{r_2}(\vec{K_2} + \mathsf{id}_{\ell+1}\vec{D}_{2,\ell+1}),\ \vec{D}_{2,j} \leftarrow \tilde{r_2}\vec{D}_{2,j} \text{ for } \ell+2 \leq j \leq h.$$

thus setting randomisers $r_1 = r_1 + \tilde{r_1}r_2$ and then $r_2 = \tilde{r_2}r_2$ for the new secret key given by $\mathcal{SK}_{\mathbf{id}:\mathsf{id}_{\ell+1}} = (\vec{K_1}, \vec{D}_{1,\ell+2}, \ldots, \vec{D}_{1,h}, \vec{K_2}, \vec{D}_{2,\ell+2}, \ldots, \vec{D}_{2,h})$.

$\mathsf{Decrypt}$ $(\mathcal{C}, \mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{SK}_{\mathbf{id}}, \mathcal{PP})$: Decryption is done as: $M = C_0/\bar{e}(\vec{C_1}, \vec{K_1})$.

4.2 Ideas Underlying the Security Proof

The strategy used for proving security for *DPVS-HIBE* will not be applicable to *DPVS-AHIBE*. This is mainly because of the presence of two copies of the identity-hash in the key - one in $\vec{K_1}$ and the other in $\vec{K_2}$. If the same strategy is followed, during the k-th step in the second reduction, the k-th key is either normal or semi-functional depending on whether the instance is real or random. If it is semi-functional, then the sets of randomisers for generating $\vec{K_1}$ and $\vec{K_2}$ have to appear properly distributed to the adversary during simulation. For the non-anonymous HIBE (i.e., *DPVS-HIBE*) there is only one such set and these are generated by appropriately randomising the elements from the instance (of DS2). But in case of *DPVS-AHIBE*, the only way to construct the second set of randomisers (without changing the bases) would be by multiplying the first set of randomisers by a common multiplier. Clearly, this results in improper distribution of the semi-functional components thus giving the adversary maximum advantage in winning the game. There are two ways to overcome this problem. One possibility is to define the semi-functional key in such a way that the second set of randomisers are obtained by multiplying the first set with a common scalar. This, however, affects our third stage of the reduction where we provide an information theoretic argument based on a change-of-bases transformation. It is natural to question whether it is possible to modify the third reduction accordingly, but we think it is unlikely. The reason is that, we need to argue about the statistical indistinguishability of the component containing the id-hash from random element of the subspace spanned by the same component. A subspace assumption or a its variant cannot be used for such an argument. The other way to get around the problem is to use two separate instances to generate the two parts of the key so that they have the proper distribution when semi-functional. In order to do this, we need to change the key from normal to semi-functional in two steps - first to partial semi-functional and then to semi-functional.

The first reduction is similar to that of the previous HIBE scheme. In the last stage, we use an information theoretic argument similar to [CLL$^+$12]. We modify the structure of the second stage of the proof by altering the game sequence to accommodate the partial semi-functional keys. The complete proof of security is provided in Section C.

5 Conclusion

We have presented six new constructions of efficient HIBE schemes with constant-size ciphertexts that can be instantiated with Type-3 pairings, each of them being fully secure under static assumptions. Three of these schemes have the anonymity property and within the dual system framework, these are the only

known anonymous HIBE schemes based on prime-order pairings. We explore thoroughly the area of CC-HIBE schemes and the results we obtain more or less complete the picture in this domain.

References

[ABC+05] Michel Abdalla, Mihir Bellare, Dario Catalano, Eike Kiltz, Tadayoshi Kohno, Tanja Lange, John Malone-Lee, Gregory Neven, Pascal Paillier, and Haixia Shi. Searchable encryption revisited: Consistency properties, relation to anonymous IBE, and extensions. In Victor Shoup, editor, *CRYPTO*, volume 3621 of *Lecture Notes in Computer Science*, pages 205–222. Springer, 2005.

[BBG05] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity-based encryption with constant size ciphertext. In Cramer [Cra05], pages 440–456. Full version available at Cryptology ePrint Archive; Report 2005/015.

[BF03] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. *SIAM J. Comput.*, 32(3):586–615, 2003. Earlier version appeared in the proceedings of CRYPTO 2001.

[BW06] Xavier Boyen and Brent Waters. Anonymous hierarchical identity-based encryption (without random oracles). In Cynthia Dwork, editor, *CRYPTO*, volume 4117 of *Lecture Notes in Computer Science*, pages 290–307. Springer, 2006.

[CHKM10] Sanjit Chatterjee, Darrel Hankerson, Edward Knapp, and Alfred Menezes. Comparing two pairing-based aggregate signature schemes. *Des. Codes Cryptography*, 55(2-3):141–167, 2010.

[CLL+12] Jie Chen, Hoon Wei Lim, San Ling, Huaxiong Wang, and Hoeteck Wee. Shorter IBE and signatures via asymmetric pairings. *IACR Cryptology ePrint Archive*, 2012:224, 2012.

[CM11] Sanjit Chatterjee and Alfred Menezes. On cryptographic protocols employing asymmetric pairings – the role of ψ revisited. *Discrete Applied Mathematics*, 159(13):1311–1322, 2011.

[Coc01] Clifford Cocks. An identity-based encryption scheme based on quadratic residues. In Bahram Honary, editor, *IMA Int. Conf.*, volume 2260 of *Lecture Notes in Computer Science*, pages 360–363. Springer, 2001.

[Cra05] Ronald Cramer, editor. *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *Lecture Notes in Computer Science*. Springer, 2005.

[CS06] Sanjit Chatterjee and Palash Sarkar. New constructions of constant size ciphertext hibe without random oracle. In M.S. Rhee and B. Lee, editors, *ICISC*, volume 4296 of *Lecture Notes in Computer Science*, pages 310–327. Springer, 2006.

[CS07] Sanjit Chatterjee and Palash Sarkar. Constant size ciphertext hibe in the augmented selective-id model and its extensions. *J. UCS*, 13(10):1367–1395, 2007.

[DCIP10] Angelo De Caro, Vincenzo Iovino, and Giuseppe Persiano. Fully secure anonymous hibe and secret-key anonymous ibe with short ciphertexts. In Marc Joye, Atsuko Miyaji, and Akira Otsuka, editors, *Pairing-Based Cryptography - Pairing 2010*, volume 6487 of *Lecture Notes in Computer Science*, pages 347–366. Springer Berlin / Heidelberg, 2010.

[Duc10] Léo Ducas. Anonymity from asymmetry: New constructions for anonymous hibe. In Josef Pieprzyk, editor, *CT-RSA*, volume 5985 of *Lecture Notes in Computer Science*, pages 148–164. Springer, 2010.

[GPS08] Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, 2008.

[GS02] Craig Gentry and Alice Silverberg. Hierarchical ID-based cryptography. In Yuliang Zheng, editor, *ASIACRYPT*, volume 2501 of *Lecture Notes in Computer Science*, pages 548–566. Springer, 2002.

[HL02] Jeremy Horwitz and Ben Lynn. Toward hierarchical identity-based encryption. In Lars R. Knudsen, editor, *EUROCRYPT*, volume 2332 of *Lecture Notes in Computer Science*, pages 466–481. Springer, 2002.

[Lew12] Allison B. Lewko. Tools for simulating features of composite order bilinear groups in the prime order setting. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT*, volume 7237 of *Lecture Notes in Computer Science*, pages 318–335. Springer, 2012.

[LW10] Allison B. Lewko and Brent Waters. New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In Daniele Micciancio, editor, *TCC*, volume 5978 of *Lecture Notes in Computer Science*, pages 455–479. Springer, 2010.

[OT08] Tatsuaki Okamoto and Katsuyuki Takashima. Homomorphic encryption and signatures from vector decomposition. In Steven D. Galbraith and Kenneth G. Paterson, editors, *Pairing*, volume 5209 of *Lecture Notes in Computer Science*, pages 57–74. Springer, 2008.

[OT09] Tatsuaki Okamoto and Katsuyuki Takashima. Hierarchical predicate encryption for inner-products. In Mitsuru Matsui, editor, *ASIACRYPT*, volume 5912 of *Lecture Notes in Computer Science*, pages 214–231. Springer, 2009.

[Sha84] Adi Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and David Chaum, editors, *CRYPTO*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer, 1984.

[SKOS09] Jae Hong Seo, Tetsutaro Kobayashi, Miyako Ohkubo, and Koutarou Suzuki. Anonymous hierarchical identity-based encryption with constant size ciphertexts. In Stanislaw Jarecki and Gene Tsudik, editors, *Public Key Cryptography*, volume 5443 of *Lecture Notes in Computer Science*, pages 215–234. Springer, 2009.

[SV07] Nigel P. Smart and Frederik Vercauteren. On computable isomorphisms in efficient asymmetric pairing-based systems. *Discrete Applied Mathematics*, 155(4):538–547, 2007.

[Wat09] Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In Shai Halevi, editor, *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 619–636. Springer, 2009.

A Rest of the Preliminaries

A.1 Dual Pairing Vector Spaces

Definition A.1. A dual pairing vector space (DPVS) $\mathcal{V}_n$ is a tuple $(p, \mathbb{V}, \mathbb{V}^*, \mathbb{G}_T, \bar{e}, \mathbb{A}, \mathbb{A}^*)$ where $\mathbb{V}$, $\mathbb{V}^*$ are n-dimensional vector spaces over $\mathbb{Z}_p$, $\mathbb{G}_T$ is a cyclic group of order p and $\mathbb{A} = (\vec{A}_1, \ldots, \vec{A}_n)$, $\mathbb{A}^* = (\vec{A}_1^*, \ldots, \vec{A}_n^*)$ are the canonical bases of $\mathbb{V}$ and $\mathbb{V}^*$ respectively and the following conditions are satisfied.

- There exists an efficiently computable non-degenerate bilinear map $\bar{e} : \mathbb{V} \times \mathbb{V}^* \to \mathbb{G}_T$. For $r, s \in \mathbb{Z}_p$, $\bar{e}(r\vec{P}, s\vec{Q}) = \bar{e}(\vec{P}, \vec{Q})^{rs}$ and if $\bar{e}(\vec{P}, \vec{Q}) = 1$ for all $\vec{P} \in \mathbb{V}$, then $\vec{Q} = \vec{0}$.
- For all i, j, $\bar{e}(\vec{A}_i, \vec{A}_j^*) = g_T^{\delta_{i,j}}$, where $\delta_{i,j}$ is the Kronecker delta function defined by $\delta_{i,j} = 1$ if $i = j$ and 0 otherwise. Here $g_T \in \mathbb{G}_T$ and $g_T \neq 1$.
- There exist endomorphisms $\phi_{i,j}$ (resp. $\phi_{i,j}^*$) of $\mathbb{V}$ (resp. $\mathbb{V}^*$) such that $\phi_{i,j}(\vec{A}_j) = \vec{A}_i$ (resp. $\phi_{i,j}^*(\vec{A}_j^*) = \vec{A}_i^{\,*}$) and for $k \neq j$, $\phi_{i,j}(\vec{A}_k) = \vec{0}$ (resp. $\phi_{i,j}^*(\vec{A}_k^*) = \vec{0}$). $\phi_{i,j}$ and $\phi_{i,j}^*$ are efficiently computable and are called *distortion maps*.

A.2 HIBE - Definition and Security Models

A HIBE scheme consists of five probabilistic polynomial time algorithms - Setup , Encrypt , KeyGen , Delegate and Decrypt .

- Setup based on an input security parameter λ, generates and outputs the public parameters $\mathcal{PP}$ and the master secret $\mathcal{MSK}$.
- KeyGen inputs an identity vector $\mathbf{id}$ and master secret $\mathcal{MSK}$ and outputs the secret key $\mathcal{SK}_{\mathbf{id}}$ corresponding to $\mathbf{id}$.
- Encrypt inputs an identity $\mathbf{id}$, a message M and returns a ciphertext $\mathcal{C}$.
- Delegate takes as input a depth ℓ identity vector $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$, a secret key $\mathcal{SK}_{\mathbf{id}}$ and an identity $\mathsf{id}_{\ell+1}$; returns a secret key for the identity vector $(\mathsf{id}_1, \ldots, \mathsf{id}_{\ell+1})$.
- Decrypt inputs a ciphertext $\mathcal{C}$, an identity vector $\mathbf{id}$, secret key $\mathcal{SK}_{\mathbf{id}}$ and returns either the corresponding message M or $\perp$ indicating failure.

A.2.1 Security against Chosen Plaintext Attack

The notion of security for HIBE we use is indistinguishability of ciphertexts under a chosen plaintext attack (CPA) defined in [GS02]. It is modelled as a game between an adversary $\mathscr{A}$ and a simulator which we denote as ind-cpa. The following are the different phases of the game.

Setup: The challenger runs the Setup algorithm of the HIBE and gives the public parameters to $\mathscr{A}$.

Phase 1: $\mathscr{A}$ makes a number of key extraction queries adaptively. For a query for an identity vector $\mathbf{id}$, the challenger responds with a key $\mathcal{SK}_{\mathbf{id}}$.

Challenge: $\mathscr{A}$ provides two messages pairs M_0, M_1 and a challenge identity $\widehat{\mathbf{id}}$ such that neither $\widehat{\mathbf{id}}$ nor any of their prefixes are queried in **Phase 1**. The challenger then chooses a bit β uniformly at random from $\{0,1\}$ and returns an encryption $\widehat{\mathcal{C}}$ of M_β under $\widehat{\mathbf{id}}$ to $\mathscr{A}$.

Phase 2: $\mathscr{A}$ issues more key extraction queries as in **Phase 1** with the restriction that no queried identity $\mathbf{id}$ is a prefix of $\widehat{\mathbf{id}}$.

Guess: $\mathscr{A}$ outputs a bit β'.

If $\beta = \beta'$, then $\mathscr{A}$ wins the game. The advantage of $\mathscr{A}$ in breaking the security of the HIBE scheme in the ind-cpa game is given by

$$\mathsf{Adv}^{\mathsf{ind-cpa}}_{\mathrm{HIBE}}(\mathscr{A}) = \left| \Pr[\beta = \beta'] - \frac{1}{2} \right|.$$

The HIBE scheme is said to be (ε, t, q)-IND-ID-CPA secure (secure under chosen plaintext attack) if for every adversary $\mathscr{A}$ making at most q queries and whose running time is t, it holds that $\mathsf{Adv}^{\mathsf{ind-cpa}}_{\mathrm{HIBE}}(\mathscr{A}) \leq \varepsilon$.

A.2.2 Anonymous HIBE

The following security game captures both anonymity and security against a chosen plaintext attack for a HIBE scheme and will be called ano-ind-cpa.

Setup: The challenger runs the Setup algorithm of the HIBE and gives the public parameters to $\mathscr{A}$.

Phase 1: $\mathscr{A}$ makes a number of key extraction queries adaptively. For a query for an identity vector $\mathbf{id}$, the challenger responds with a key $\mathcal{SK}_{\mathbf{id}}$.

Challenge: $\mathscr{A}$ provides two message-identity pairs $(M_0, \widehat{\mathbf{id}}_0)$ and $(M_1, \widehat{\mathbf{id}}_1)$ as challenge with the restriction that neither $\widehat{\mathbf{id}}_0, \widehat{\mathbf{id}}_1$ nor any of their prefixes should have been queried in **Phase 1**. The challenger then chooses a bit β uniformly at random from $\{0,1\}$ and returns an encryption $\widehat{\mathcal{C}}$ of M_β under the identity $\widehat{\mathbf{id}}_\beta$ to $\mathscr{A}$.

Phase 2: $\mathscr{A}$ issues more key extraction queries as in **Phase 1** with the restriction that no queried identity $\mathbf{id}$ is a prefix of $\widehat{\mathbf{id}}_\beta$.

Guess: $\mathscr{A}$ outputs a bit β'.

The advantage of $\mathscr{A}$ in breaking the security of the HIBE scheme in the game ano-ind-cpa given by

$$\mathsf{Adv}^{\mathsf{ano-ind-cpa}}_{\mathrm{HIBE}}(\mathscr{A}) = \left| \Pr[\beta = \beta'] - \frac{1}{2} \right|.$$

The HIBE scheme is said to be (ε, t, q)-ANO-IND-ID-CPA secure if every t-time adversary making at most q queries has $\mathsf{Adv}^{\mathsf{ano-ind-cpa}}_{\mathrm{HIBE}}(\mathscr{A}) \leq \varepsilon$.

A.3 Complexity Assumptions

Here we state some hardness assumptions in asymmetric bilinear maps that are used in security reductions. Let $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, P_1, P_2)$ be an asymmetric pairing and $\mathcal{V}_n = (p, \mathbb{V}, \mathbb{V}^*, \mathbb{G}_T, \bar{e}, \mathbb{A}, \mathbb{A}^*)$ be a DPVS of dimension n over $\mathcal{G}$.

Assumption LW1. Let $F_1 \in \mathbb{G}_1$ and $F_2 \in \mathbb{G}_2$ be random generators and let $a, b, s \in_{\mathrm{U}} \mathbb{Z}_p$. Let $Y_1 \in_{\mathrm{U}} \mathbb{G}_1$. For a probabilistic polynomial time (PPT) algorithm $\mathscr{A}$ that outputs 0 or 1, its advantage in solving the

LW1 problem is given by

$$\begin{aligned}\mathsf{Adv}_{\mathcal{G}}^{\mathrm{LW1}}(\mathscr{A}) =& |\Pr[\mathscr{A}(\mathcal{G}, F_1, bsF_1, sF_1, aF_1, ab^2F_1, bF_1, b^2F_1, asF_1, b^2sF_1, b^3F_1, b^3sF_1, F_2, bF_2, ab^2sF_1) = 1] \\ & - \Pr[\mathcal{G}, \mathscr{A}(F_1, bsF_1, sF_1, aF_1, ab^2F_1, bF_1, b^2F_1, asF_1, b^2sF_1, b^3F_1, b^3sF_1, F_2, bF_2, Y_1) = 1]|\end{aligned}$$

The (ε, t)-LW1 assumption holds in $\mathcal{G}$ if for any adversary $\mathscr{A}$ running in time at most t, $\mathsf{Adv}_{\mathcal{G}}^{\mathrm{LW1}}(\mathscr{A}) \leq \varepsilon$.

Assumption LW2. Let $F_1 \in \mathbb{G}_1$ and $F_2 \in \mathbb{G}_2$ be random generators. Let d, x_1, x_2 denote random elements of $\mathbb{Z}_p$. The LW2 problem is to decide, given

$$(\mathcal{G}, F_1, dF_1, d^2F_1, bxF_1, dbxF_1, d^2xF_1, F_2, dF_2, bF_2, cF_2, Z_2),$$

whether $Z_2 = bcF_2$ or $Z_2 \in_{\mathrm{U}} \mathbb{G}_2$. The (ε, t)-LW2 assumption is that any t-time algorithm $\mathscr{A}$ has advantage at most ε in solving the LW2 problem in $\mathcal{G}$.

Decisional Bilinear Diffie-Hellman (DBDH) assumption. Let P_1, P_2 be random generators of $\mathbb{G}_1$,$\mathbb{G}_2$ respectively and let x_1, x_2, x_3 be random elements of $\mathbb{Z}_p$. Let Y_T be a random element of $\mathbb{G}_T$. For a PPT algorithm $\mathscr{A}$ that outputs either 0 or 1, define its advantage in solving the DBDH problem as follows.

$$\begin{aligned}\mathsf{Adv}_{\mathcal{G}}^{\mathrm{DBDH}}(\mathscr{A}) =& |\Pr[\mathscr{A}(\mathcal{G}, P_1, x_1P_1, x_2P_1, x_3P_1, P_2, x_1P_2, x_2P_2, x_3P_2, e(P_1, P_2)^{x_1x_2x_3}) = 1] \\ & - \Pr[\mathscr{A}(\mathcal{G}, P_1, x_1P_1, x_2P_1, x_3P_1, P_2, x_1P_2, x_2P_2, x_3P_2, Y_T) = 1]|\end{aligned}$$

We say that the (ε, t)-DBDH assumption holds in $\mathcal{G}$ if $\mathsf{Adv}_{\mathcal{G}}^{\mathrm{DBDH}}(\mathscr{A}) \leq \varepsilon$ for every algorithm $\mathscr{A}$ running in time at most t.

Assumption A1. Let $F_1 \in \mathbb{G}_1$ and $F_2 \in \mathbb{G}_2$ be random generators. Let x, d, s denote random elements of $\mathbb{Z}_p$. The A1 problem is to decide, given

$$(\mathcal{G}, F_1, dxF_1, sdF_1, F_2, dxF_2, Z_1),$$

whether $Z_1 = sdxF_2$ or $Z_1 \in_{\mathrm{U}} \mathbb{G}_1$. The (ε, t)-A1 assumption is that any t-time algorithm $\mathscr{A}$ has advantage at most ε in solving the A1 problem in $\mathcal{G}$.

Decision Diffie-Hellman (DDH) Let $x_1, x_2 \in_{\mathrm{U}} \mathbb{Z}_p$, $Q_1(\neq 0) \in \mathbb{G}_1$, $Q_2(\neq 0) \in \mathbb{G}_2$. The DDH problem in $\mathbb{G}_1$ (DDH1) is to decide, given $(\mathcal{G}, Q_1, x_1Q_1, x_2Q_1, Q_2, Z_1)$, whether $Z_1 = x_1x_2Q_1$ or Z_1 is a random element of $\mathbb{G}_1$.

Let $\mathscr{A}$ be a probabilistic polynomial time (PPT) algorithm that outputs either 0 or 1. Define its advantage in solving the DDH1 as follows.

$$\mathsf{Adv}_{\mathcal{G}}^{\mathrm{DDH1}}(\mathscr{A}) = |\Pr[\mathscr{A}(\mathcal{G}, Q_1, x_1Q_1, x_2Q_1, Q_2, x_1x_2Q_1) = 1] - \Pr[\mathscr{A}(\mathcal{G}, Q_1, x_1Q_1, x_2Q_1, Q_2, Y_1) = 1]|$$

where $Y_1 \in_{\mathrm{U}} \mathbb{G}_1$. We say that (ε, t)-DDH1 assumption holds in $\mathcal{G}$ if every algorithm $\mathscr{A}$ that runs in time less than or equal to t has $\mathsf{Adv}_{\mathcal{G}}^{\mathrm{DDH1}}(\mathscr{A}) \leq \varepsilon$. Similarly one can define DDH assumption in $\mathbb{G}_2$ (DDH2).

Decisional Subspace Given a DPVS $\mathcal{V}_n$, define a tuple $\mathcal{D}$ constructed according to the following distribution.

$$(\mathbf{D}, \mathbf{D}^*) \longleftarrow \mathsf{Dual}(\mathbb{Z}_p^n)$$

$$\tau_1, \tau_2, \mu_1, \mu_2 \xleftarrow{\mathrm{U}} \mathbb{Z}_p,$$

$$\vec{U}_1 = (\mu_1 \mathbf{d}_1^* + \mu_2 \mathbf{d}_{k+1}^*)P_2, \vec{U}_2 = (\mu_1 \mathbf{d}_2^* + \mu_2 \mathbf{d}_{k+2}^*)P_2, \ldots, \vec{U}_k = (\mu_1 \mathbf{d}_k^* + \mu_2 \mathbf{d}_{2k}^*)P_2,$$

$$\vec{V}_1 = \tau_1 \mathbf{d}_1 P_1, \vec{V}_2 = \tau_1 \mathbf{d}_2 P_1, \ldots, \vec{V}_k = \tau_1 \mathbf{d}_k P_1,$$

$$\vec{W}_1 = (\tau_1 \mathbf{d}_1 + \tau_2 \mathbf{d}_{k+1})P_1, \vec{W}_2 = (\tau_1 \mathbf{d}_2 + \tau_2 \mathbf{d}_{k+2})P_1, \ldots, \vec{W}_k = (\tau_1 \mathbf{d}_k + \tau_2 \mathbf{d}_{2k})P_1,$$

$$\mathcal{D} = (\mathcal{G}, \mathbf{d}_1 P_1, \ldots, \mathbf{d}_n P_1, \mathbf{d}_1^* P_2, \ldots, \mathbf{d}_k^* P_2, \mathbf{d}_{2k+1}^* P_2, \ldots, \mathbf{d}_n^* P_2, \vec{U}_1, \ldots, \vec{U}_k, \mu_2),$$

where k, n are positive integers with $2k \leq n$. The decisional subspace problem in $\mathbb{V}$ is to distinguish between $(\vec{V}_1, \ldots, \vec{V}_k)$ and $(\vec{W}_1, \ldots, \vec{W}_k)$ given $\mathcal{D}$.

For a PPT algorithm $\mathscr{A}$ that outputs 0 or 1, its advantage in solving DS1 problem is defined as

$$\mathsf{Adv}_{\mathcal{V}_n}^{\mathrm{DS1}}(\mathscr{A}) = |\Pr[\mathscr{A}(\mathcal{D}, \vec{V}_1, \ldots, \vec{V}_k) = 1] - \Pr[\mathscr{A}(\mathcal{D}, \vec{W}_1, \ldots, \vec{W}_k) = 1]|.$$

The (ε, t)-DS1 assumption holds in $\mathcal{V}_n$ if for every algorithm $\mathscr{A}$ running in time at most t, we have $\mathsf{Adv}_{\mathcal{V}_n}^{\mathrm{DS1}}(\mathscr{A}) \leq \varepsilon$. Similarly one can define the decisional subspace assumption in $\mathbb{V}^*$ (DS2). Note that the problem is easy when $\mathbf{D}, \mathbf{D}^*$ are the canonical bases $\mathbf{A}, \mathbf{A}^*$.

Reductions Let $\mathcal{V}_n$ be a DPVS obtained from an asymmetric pairing $\mathcal{G}$. [CLL$^+$12] provides a reduction from the DDH1(resp. DDH2) problems in $\mathcal{G}$ to the DS1(resp. DS2) problems in $\mathcal{V}_n$. The lemma below summarises the result.

Lemma A.1. *If the* (ε, t)-DDH1 *(resp.* (ε, t)-DDH2*) assumption holds in* $\mathcal{G}$ *, then the subspace assumption* (ε', t')-DS1 *(resp.* (ε', t')-DS2*) holds in* $\mathcal{V}_n$ *where* $\varepsilon = \varepsilon'$*,* $t = t' + O(nk\rho)$ *and* ρ *is the maximum time required for one scalar multiplication in either* $\mathbb{G}_1$ *or* $\mathbb{G}_2$*.*

B Proof of Security of *DPVS-HIBE*

The security of this HIBE follows the dual system approach. This requires the definition of semi-functional ciphertext and keys. The modification of a normal ciphertext to obtain a semi-functional ciphertext is exactly the same as in [CLL$^+$12]. For the semi-functional keys, the modification of the component $\vec{K}$ of a normal key to obtain the semi-functional counterpart is same as that in [CLL$^+$12]. The extra components required for delegation are not present in [CLL$^+$12] and we provide the definitions of the corresponding semi-functional components.

Semi-functional ciphertext: Let $(C_0, \vec{C}_1)$ be a normal ciphertext for some message M and identity vector $\mathbf{id}$. Semi-functional ciphertext is obtained by modifying $\vec{C}_1$ as follows:

$$t_1, t_2 \xleftarrow{\mathrm{U}} \mathbb{Z}_p,$$

$$\vec{C}_1 \leftarrow \vec{C}_1 + (t_1 \mathbf{b}_3 + t_2 \mathbf{b}_4)P_1.$$

Semi-functional key: For an identity vector $\mathbf{id}$ of length ℓ, let $(\vec{K}, \vec{D}_{\ell+1}, \ldots, \vec{D}_h)$ be a normal key. A semi-functional key for $\mathbf{id}$ is computed as follows:

$$z_1, z_2, x_{\ell+1}, \ldots, x_h, y_{\ell+1}, \ldots, y_h \xleftarrow{\mathrm{U}} \mathbb{Z}_p$$

$$\vec{K} \leftarrow \vec{K} + (z_1 \mathbf{b}_3^* + z_2 \mathbf{b}_4^*) P_2$$

$$\vec{D}_j \leftarrow \vec{D}_j + x_j \mathbf{b}_3^* P_2 + y_j \mathbf{b}_4^* P_2 \quad \text{for } \ell + 1 \leq j \leq h$$

The vectors $\mathbf{b}_3, \mathbf{b}_4$ and $\mathbf{b}_3^*, \mathbf{b}_4^*$ are used only in defining the semi-functional components. The scheme itself does not use them.

Nominal Semi-functionality. Decryption of a semi-functional ciphertext $\mathcal{C}$ with a semi-functional key $\mathcal{SK}_{\mathbf{id}}$ for an identity $\mathbf{id}$ will succeed only when $t_1 z_1 + t_2 z_2 \equiv 0 \pmod{p}$. Such a pair $(\mathcal{C}, \mathcal{SK}_{\mathbf{id}})$ is called *nominally semi-functional*.

We now present two lemmas that will be useful in the reductions that follow. The first one is from [CLL+12].

Lemma B.1. *Let $\mathcal{S} = \{(\mathbf{u}, \mathbf{v}) : \mathbf{u}, \mathbf{v} \in \mathbb{Z}_p^n \text{ and } \mathbf{u} \cdot \mathbf{v} \neq 0\}$. If $\mathbf{X} \xleftarrow{\mathrm{U}} \mathbb{Z}_p^{n \times n}$ is invertible, then for all $(\mathbf{x}, \mathbf{v}) \in \mathcal{S}$, for all $(\mathbf{r}, \mathbf{w}) \in \mathcal{S}$, and for $\rho, \tau \xleftarrow{\mathrm{U}} \mathbb{Z}_p^\times$,*

$$\Pr\left[\left(\rho \mathbf{X}^{-1} \mathbf{u} = \mathbf{r}\right) \ \wedge \ \left(\tau \mathbf{X}^T \mathbf{v} = \mathbf{w}\right)\right] = \frac{1}{|\mathcal{S}|}$$

where $|\mathcal{S}| = (p^n - 1)(p^n - p^{n-1})$ and the probability is over the choice of $\mathbf{X}$, ρ and τ.

Essentially, the lemma suggests that $\rho \mathbf{U}^T \mathbf{u}$ and $\tau \mathbf{X}^T \mathbf{v}$ are uniformly and independently distributed unless $\mathbf{r} \cdot \mathbf{w} = 0$. This condition holds only when $\mathbf{x} \cdot \mathbf{v} = 0$ which happens with probability $1/p$.

Lemma B.2. *Let $\mathbf{X} \xleftarrow{\mathrm{U}} \mathbb{Z}_p^{n \times n}$. Then*

$$\Pr[\mathbf{X} \in GL(n, \mathbb{Z}_p)] = \frac{\prod_{i=1}^{n} (p^n - p^{i-1})}{p^{n^2}}.$$

Proof. Choosing $\mathbf{X} \xleftarrow{\mathrm{U}} \mathbb{Z}_p^{n \times n}$ with $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ is equivalent to picking the vectors $\mathbf{x}_i$ $(1 \leq i \leq n)$ uniformly and independently from $\mathbb{Z}_p^n$. There are a total of p^{n^2} choices for $\mathbf{X}$. If these vectors are linearly independent, then $\mathbf{X} \in GL(n, \mathbb{Z}_p)$. The vector $\mathbf{x}_1$ has $p^n - 1$ choices (excluding $\mathbf{0}$, the all-zero vector). For the vectors to be linearly independent, $\mathbf{x}_i$ (for $2 \leq i \leq n$) must be chosen such that $\mathbf{x}_i \notin \mathsf{span}\langle \mathbf{x}_1, \ldots, \mathbf{x}_{i-1} \rangle$ and so there are $p^n - p^{i-1}$ choices for $\mathbf{x}_i$. Therefore, $|GL(n, \mathbb{Z}_p)| = \prod_{i=1}^{n} (p^n - p^{i-1})$ thus proving the lemma. □

We now present the security proof for the HIBE scheme of Section 3.3, *DPVS-HIBE*. The shorthand $\mathcal{H}_\ell(\mathbf{id})$ will be used to denote the quantity $\sum_{j=1}^{\ell} \mathsf{id}_j \theta_j$ where $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$. Let $\gamma_p = \left(1 - \frac{1}{p}\right)$. The proof goes through a sequence of $q + 4$ games, where q is the number of key extract queries made by the adversary. Let Game_{actual} denote the actual IBE CPA-security game $\mathsf{ind\text{-}cpa}$. Game_{actual} is same as $\mathrm{Game}_{restricted}$ except that the following restriction is added to the game: $\mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \not\equiv \mathcal{H}_{\ell_k}(\mathbf{id}_k) \pmod{p}$ for all $k \in \{1, \ldots, q\}$ where $\widehat{\mathbf{id}}$ is the challenge identity and $\mathbf{id}_k$ is the identity provided by the adversary in the k-th key extract query. Here $\widehat{\ell}$ and ℓ_k are the lengths of the identity tuples $\widehat{\mathbf{id}}$ and $\mathbf{id}_k$. Game_0 is just like

$\text{Game}_{restricted}$ except that the challenge ciphertext is a semi-functional encryption of the chosen message to the corresponding identity vector. Let q be the number of key extraction queries made by the adversary during the attack. Define Game_k for $1 \leq k \leq q$ such that the first k keys returned to the adversary are semi-functional and the rest are normal. Let Game_{final} be defined similar to Game_q except that now the challenge ciphertext is a semi-functional encryption of a random message to a random identity vector. Let X_{actual}, $X_{restricted}$, X_k and X_{final} denote the events that the adversary wins in Game_{actual}, $\text{Game}_{restricted}$, Game_k and Game_{final} for $0 \leq k \leq q$ respectively. Note that, in Game_{final}, the challenge ciphertext is an encryption of a random message and hence bit β is statistically hidden from the adversary's view implying that $\Pr[X_{final}] = 1/2$.

The first stage of the reduction (Game_{actual} to Game_0) is same as in [CLL+12]. Second reduction is similar with additional capabilities for handling components of the key used for delegation. In the last stage (Game_q to Game_{final}), we obtain a reduction from the DS1 assumption.

Lemma B.3. $|\Pr[X_{actual}] - \Pr[X_{restricted}]| \leq q/p$.

Proof. Call the event that $\mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \equiv \mathcal{H}_{\ell_k}(\mathbf{id}_k) \pmod p$ for some k as F. The two games Game_{actual} and $\text{Game}_{restricted}$ proceed the same way unless F occurs. For a fixed k, $\Pr[\mathsf{F}] = 1/p$. Since there are q possible values of k, we have

$$|\Pr[X_{actual}] - \Pr[X_{restricted}]| \leq q \cdot \Pr[\mathsf{F}] \leq \frac{q}{p}.$$

□

Lemma B.4. *If there exists a adversary $\mathscr{A}$ against DPVS-HIBE making at most q key extraction queries, then we can build an algorithm $\mathscr{B}$ having advantage at least $\gamma_p^2 |\Pr[X_{restricted}] - \Pr[X_0]|$ in solving the* DS1 *problem with $n = 4, k = 2$.*

Proof. The algorithm $\mathscr{B}$ receives $(\mathcal{G}, \mathbf{d}_1 P_1, \ldots, \mathbf{d}_4 P_1, \mathbf{d}_1^* P_2, \mathbf{d}_2^* P_2, \vec{U}_1, \vec{U}_2, \mu_2, \vec{T}_1, \vec{T}_2)$ as an instance of DS1 problem. Here $\vec{U}_1 = (\mu_1 \mathbf{d}_1^* + \mu_2 \mathbf{d}_3^*) P_2$ and $\vec{U}_2 = (\mu_1 \mathbf{d}_2^* + \mu_2 \mathbf{d}_4^*) P_2$. If $\vec{T}_1, \vec{T}_2$ are distributed as $\tau_1 \mathbf{d}_1 P_1, \tau_1 \mathbf{d}_2 P_1$, we call them "real" and when their distribution is $(\tau_1 \mathbf{d}_1 + \tau_2 \mathbf{d}_3) P_1, (\tau_1 \mathbf{d}_2 + \tau_2 \mathbf{d}_4) P_1$, we call them "random". So $\mathscr{B}$'s task is to decide whether $\vec{T}_1, \vec{T}_2$ are real or random and accordingly return a bit 0 or 1, respectively. We describe how $\mathscr{B}$ will simulate each phase of the security game.

Setup: $\mathscr{B}$ first chooses $\mathbf{X} \xleftarrow{\text{U}} \mathbb{Z}_p^{2\times 2}$ and if $\mathbf{X} \notin GL(2, \mathbb{Z}_p)$, aborts the game and returns a random bit. Otherwise, it sets the dual bases $(\mathbf{B}, \mathbf{B}^*)$ of the HIBE scheme as follows.

$$\mathbf{b}_1 = \mathbf{d}_1, \;\; \mathbf{b}_2 = \mathbf{d}_2, \;\; (\mathbf{b}_3, \mathbf{b}_4) = (\mathbf{d}_3, \mathbf{d}_4)\mathbf{X},$$

$$\mathbf{b}_1^* = \mathbf{d}_1^*, \;\; \mathbf{b}_2 = \mathbf{d}_2^*, \;\; (\mathbf{b}_3^*, \mathbf{b}_4^*) = (\mathbf{d}_3^*, \mathbf{d}_4^*)(\mathbf{X}^{-1})^T.$$

It then chooses $\alpha, \theta_1, \ldots, \theta_h \xleftarrow{\text{U}} \mathbb{Z}_p$ and provides the following public parameters to $\mathscr{A}$.

$$\mathcal{PP} = (\mathcal{G}, \mathbf{b}_1 P_1, \theta_1 \mathbf{b}_2 P_1, \ldots, \theta_h \mathbf{b}_2 P_1, e(P_1, P_2)^{\alpha \mathbf{b}_1 \cdot \mathbf{b}_1^*}).$$

$\mathscr{B}$ knows the master secret key $\mathcal{MSK} = (\alpha \mathbf{b}_1^* P_2, \theta_1 \mathbf{b}_1^* P_2, \ldots, \theta_h \mathbf{b}_1^* P_2, \mathbf{b}_2^* P_2)$ using which normal keys for any identity vector can be created. However, with no knowledge of $\mathbf{b}_3^*, \mathbf{b}_4^*$, it cannot create semi-functional keys.

Phase 1: $\mathscr{A}$ makes a number of key extract queries. $\mathscr{B}$ returns a normal key generated using the KeyGen algorithm for every key extract query made by $\mathscr{A}$.

Challenge: $\mathscr{B}$ receives the two messages M_0, M_1 and challenge identity $\widehat{\mathbf{id}} = (\widehat{\mathsf{id}}_1, \ldots, \widehat{\mathsf{id}}_{\widehat{\ell}})$ from $\mathscr{A}$. It chooses $\beta \xleftarrow{\text{U}} \{0,1\}$ and encrypts M_β to $\widehat{\mathbf{id}}$ as:

$$C_0 = M_\beta \cdot \bar{e}(\vec{T}_1, \mathbf{b}_1^* P_2)^\alpha, \qquad \vec{C}_1 = \vec{T}_1 + \left(\sum_{j=1}^{\widehat{\ell}} \widehat{\mathsf{id}}_j \theta_j \right) \vec{T}_2,$$

implicitly setting $s = \tau_1$. Ciphertext $\widehat{\mathcal{C}} = (C_0, \vec{C}_1)$ is returned to $\mathscr{A}$.

Now if $\vec{T}_1, \vec{T}_2$ are real, $\widehat{\mathcal{C}}$ is distributed as a normal ciphertext and in this case $\mathscr{B}$ simulates $\text{Game}_{restricted}$. Otherwise, $\widehat{\mathcal{C}}$ will be semi-functional and $\mathscr{B}$ simulates Game_0. The computation shown below illustrates this fact. Suppose that $\vec{T}_1 = \tau_1 \mathbf{d}_1 P_1$ and $\vec{T}_2 = \tau_1 \mathbf{d}_2 P_1$. Then

$$\begin{aligned} \vec{C}_1 &= \vec{T}_1 + \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \vec{T}_2 \\ &= \tau_1 \mathbf{d}_1 P_1 + \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \tau_1 \mathbf{d}_2 P_1 \\ &= s \mathbf{b}_1 P_1 + s \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \mathbf{b}_2 P_1. \end{aligned}$$

Next consider the case when $\vec{T}_1 = (\tau_1 \mathbf{d}_1 + \tau_2 \mathbf{d}_3) P_1$ and $\vec{T}_2 = (\tau_1 \mathbf{d}_2 + \tau_2 \mathbf{d}_4) P_1$. We have

$$\begin{aligned} \vec{C}_1 &= \vec{T}_1 + \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \vec{T}_2 \\ &= (\tau_1 \mathbf{d}_1 + \tau_2 \mathbf{d}_3) P_1 + \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) (\tau_1 \mathbf{d}_2 + \tau_2 \mathbf{d}_4) P_1 \\ &= \tau_1 \mathbf{d}_1 P_1 + \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \tau_1 \mathbf{d}_2 P_1 + \tau_2 \mathbf{d}_3 P_1 + \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \tau_2 \mathbf{d}_4 P_1 \\ &= s \mathbf{b}_1 P_1 + s \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \mathbf{b}_2 P_1 + (\mathbf{d}_3, \mathbf{d}_4)(1, \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}))^T \tau_2 P_1 \\ &= s \mathbf{b}_1 P_1 + s \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \mathbf{b}_2 P_1 + (\mathbf{b}_3, \mathbf{b}_4) \mathbf{X}^{-1} (1, \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}))^T \tau_2 P_1, \end{aligned}$$

thus setting $(t_1, t_2) = \tau_2 \mathbf{X}^{-1} (1, \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}))^T$ which has the appropriate distribution due to the choice of $\mathbf{X}$ and the fact that all information provided to $\mathscr{A}$ are independent of $\mathbf{X}$. $\mathscr{B}$'s simulation is, therefore, perfect.

Phase 2: As in first phase, $\mathscr{B}$ returns a normal key for every query.

Guess: $\mathscr{A}$ returns its guess β'.

If the adversary wins the game then $\mathscr{B}$ returns 1; otherwise it returns 0. Let Y_{real} and Y_{random} denote the events that $\mathscr{B}$ returns 1 when $\vec{T}_1, \vec{T}_2$ are real and random respectively. Let the event abort denote the event that $\mathscr{B}$ aborts the game. Then, we have

$$\begin{aligned} \Pr[Y_{real}] &= \Pr[Y_{real} | \mathsf{abort}] \Pr[\mathsf{abort}] + \Pr[Y_{real} | \overline{\mathsf{abort}}] \Pr[\overline{\mathsf{abort}}] \\ &= \frac{1}{2} \Pr[\mathsf{abort}] + \Pr[X_{restricted}] \Pr[\overline{\mathsf{abort}}], \qquad (1) \\ \Pr[Y_{random}] &= \Pr[Y_{random} | \mathsf{abort}] \Pr[\mathsf{abort}] + \Pr[Y_{random} | \overline{\mathsf{abort}}] \Pr[\overline{\mathsf{abort}}] \\ &= \frac{1}{2} \Pr[\mathsf{abort}] + \Pr[X_0] \Pr[\overline{\mathsf{abort}}] \qquad (2) \end{aligned}$$

Recall that $\mathscr{B}$ aborts when $\mathbf{X}$ is not invertible. From B.2 it follows that $\Pr[\overline{\mathsf{abort}}] = (p^2 - 1)(p - 1)/p^3 \geq (p-1)^2/p^2 = \gamma_p^2$. Subtracting (2) from (1), we obtain

$$\begin{aligned} \Pr[Y_{real}] - \Pr[Y_{random}] &= (\Pr[X_{restricted}] - \Pr[X_0]) \Pr[\overline{\mathsf{abort}}] \\ &\geq (\Pr[X_{restricted}] - \Pr[X_0]) \gamma_p^2. \end{aligned}$$

$\mathscr{B}$'s advantage in solving the DS1 problem is given by

$$\mathsf{Adv}^{\mathrm{DS1}}_{\mathcal{V}_4}(\mathscr{B}) = |\Pr[Y_{real}] - \Pr[Y_{random}]| \geq |\Pr[X_{restricted}] - \Pr[X_0]|\gamma_p^2.$$

□

Lemma B.5. *If there exists an adversary $\mathscr{A}$ against DPVS-HIBE making at most q key extraction queries, then we can build an algorithm $\mathscr{B}$ having advantage at least $\gamma_p^2\,(|\Pr[X_{k-1}] - \Pr[X_k]|)$ in solving the* DS2 *problem with $n = 4, k = 2$.*

Proof. An instance $(\mathcal{G}, \mathbf{d}_1P_1, \mathbf{d}_2P_1, \mathbf{d}_1^*P_2, \ldots, \mathbf{d}_4^*P_2, \vec{U}_1, \vec{U}_2, \mu_2, \vec{T}_1, \vec{T}_2)$ of DS2 is provided to $\mathscr{B}$. Here $\vec{U}_1 = (\mu_1\mathbf{d}_1 + \mu_2\mathbf{d}_3)P_1$ and $\vec{U}_2 = (\mu_1\mathbf{d}_2 + \mu_2\mathbf{d}_4)P_1$. $\mathscr{B}$ has to determine whether $\vec{T}_1, \vec{T}_2$ are real or random - i.e., whether they are distributed as $\tau_1\mathbf{d}_1^*P_2, \tau_1\mathbf{d}_2^*P_2$ or $(\tau_1\mathbf{d}_1^* + \tau_2\mathbf{d}_3^*)P_2, (\tau_1\mathbf{d}_2^* + \tau_2\mathbf{d}_4^*)P_2$.

Setup: $\mathscr{B}$ samples $\mathbf{X} = (\chi_{i,j}) \xleftarrow{\mathrm{U}} \mathbb{Z}_p^{2\times 2}$. If $\mathbf{X}$ is not invertible, it aborts and returns a random bit; otherwise it proceeds to set the dual bases $(\mathbf{B}, \mathbf{B}^*)$ of the HIBE scheme as follows.

$$\mathbf{b}_1 = \mathbf{d}_1,\ \mathbf{b}_2 = \mathbf{d}_2,\ (\mathbf{b}_3, \mathbf{b}_4) = (\mathbf{d}_3, \mathbf{d}_4)\mathbf{X},$$

$$\mathbf{b}_1^* = \mathbf{d}_1^*,\ \mathbf{b}_2^* = \mathbf{d}_2^*,\ (\mathbf{b}_3^*, \mathbf{b}_4^*) = (\mathbf{d}_3^*, \mathbf{d}_4^*)(\mathbf{X}^{-1})^T.$$

$\mathscr{B}$ then picks $\alpha, \theta_1, \ldots, \theta_h \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ and sends the public parameters $\mathcal{PP}$ to $\mathscr{A}$. $\mathscr{B}$ knows $\mathcal{MSK}$ and hence can create normal keys. Knowledge of $\mathbf{b}_3^*, \mathbf{b}_4^*$ enables it to generate semi-functional keys too.

Phases 1 and 2: Let $\mathbf{id}_1, \mathbf{id}_2, \ldots, \mathbf{id}_q$ denote the identities for which $\mathscr{A}$ requests the secret keys. For $i > k$, $\mathscr{B}$ generates a normal key for $\mathbf{id}_i$ using the master secret and returns the resulting key to $\mathscr{A}$. For $i < k$, $\mathscr{B}$ first generates a normal key and then modifies it appropriately to obtain a semi-functional key. The resulting key is returned to $\mathscr{A}$. In case of the k-th query (i.e., $i = k$), $\mathscr{B}$ generates the key as follows. Let $\mathbf{id}_k = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$.

$$\vec{K} = \alpha\mathbf{b}_1^*P_2 + \mathcal{H}_\ell(\mathbf{id}_k)\vec{T}_1 - \vec{T}_2,$$
$$\vec{D}_{\ell+1} = \theta_{\ell+1}\vec{T}_1,\ \ldots,\ \vec{D}_h = \theta_h\vec{T}_1.$$

The key $\mathcal{SK}_{\mathbf{id}_k} = (\vec{K}, \vec{D}_{\ell+1}, \ldots, \vec{D}_h)$ is returned to $\mathscr{A}$.

If $\vec{T}_1, \vec{T}_2$ are real, then it is clear that $\mathcal{SK}_{\mathbf{id}_k}$ is going to be normal with r being set to τ_1 implicitly. In this case $\mathscr{B}$ simulates Game_{k-1}. Otherwise $\vec{T}_1, \vec{T}_2$ would be random and the key will be semi-functional as justified below.

$$\begin{aligned}
\vec{K} &= \alpha\mathbf{b}_1^*P_2 + \mathcal{H}_\ell(\mathbf{id}_k)\vec{T}_1 - \vec{T}_2 \\
&= \alpha\mathbf{b}_1^*P_2 + \mathcal{H}_\ell(\mathbf{id}_k)\tau_1\mathbf{d}_1^*P_2 - \tau_1\mathbf{d}_2^*P_2 + \tau_2\mathcal{H}_\ell(\mathbf{id}_k)\mathbf{d}_3^*P_2 - \tau_2\mathbf{d}_4^*P_2 \\
&= \alpha\mathbf{b}_1^*P_2 + r\mathcal{H}_\ell(\mathbf{id}_k)\mathbf{b}_1^*P_2 - r\mathbf{b}_2^*P_2 + (\mathbf{b}_3^*, \mathbf{b}_4^*)\mathbf{X}^T\tau_2(\mathcal{H}_\ell(\mathbf{id}_k), -1)^TP_2, \\
\vec{D}_j &= \theta_j\vec{T}_1 \\
&= \theta_j(\tau_1\mathbf{d}_1^* + \tau_2\mathbf{d}_3^*)P_2 \\
&= \theta_j\tau_1\mathbf{b}_1^*P_2 + \theta_j\tau_2(\chi_{1,1}\mathbf{b}_3^* + \chi_{1,2}\mathbf{b}_4^*)P_2,
\end{aligned}$$

setting $r = \tau_1$, $(z_1, z_2)^T = \tau_2\mathbf{X}^T(\mathcal{H}_\ell(\mathbf{id}_k), -1)^T$, $x_j = \theta_j\tau_2\chi_{1,1}$ and $y_j = \theta_j\tau_2\chi_{1,2}$ for $\ell + 1 \leq j \leq h$. Consequently, $\mathscr{B}$ simulates Game_k.

Challenge: $\mathscr{B}$ receives messages M_0, M_1 and challenge identity $\widehat{\mathbf{id}} = (\widehat{\mathsf{id}}_1, \ldots, \widehat{\mathsf{id}}_{\widehat{\ell}})$ from $\mathscr{A}$. It chooses $\beta \in \{0, 1\}$ at random. $\mathscr{B}$ encrypts M_β to $\widehat{\mathbf{id}}$ as

$$C_0 = M_\beta \cdot \bar{e}(\vec{U}_1, \mathbf{b}_1^*P_2)^\alpha,\ \vec{C}_1 = \vec{U}_1 + \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})\vec{U}_2$$

and returns $\widehat{\mathcal{C}} = (C_0, \vec{C}_1)$ to $\mathscr{A}$. Component $\vec{C}_1$ is properly distributed since

$$\begin{aligned}\vec{C}_1 &= \vec{U}_1 + \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})\vec{U}_2 \\ &= \mu_1\mathbf{d}_1P_1 + \mu_1\mathbf{d}_2\mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})P_1 + \mu_2\mathbf{d}_3P_1 + \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})\mu_2\mathbf{d}_4P_1 \\ &= \mu_1\mathbf{b}_1P_1 + \mu_1\mathbf{b}_2\mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})P_1 + (\mathbf{b}_3, \mathbf{b}_4)\mathbf{X}^{-1}\mu_2(1, \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}))^TP_1\end{aligned}$$

implicitly setting $s = \mu_1$ and $(t_1, t_2)^T = \mu_2\mathbf{X}^{-1}(1, \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}))^T$.

Guess: $\mathscr{A}$ returns its guess β' of β.

We first argue that all information provided to $\mathscr{A}$ have the right distribution. This is straightforward in case of Game_{k-1}. Due to the choice of the θ_j's and $\mathbf{X}$, the scalars x_j, y_j (for $\ell + 1 \le j \le h$) will be uniformly distributed in $\mathbb{Z}_p$. Therefore the semi-functional components of $\vec{D}_j$'s will be properly distributed. Next we show that in Game_k, the semi-functional components of $\mathcal{SK}_{\mathbf{id}_k}$ and $\widehat{\mathcal{C}}$ are properly distributed in $\mathscr{A}$'s view. For this, we need only argue that the co-efficient vectors (t_1, t_2) and (z_1, z_2) given by

$$\begin{aligned}(t_1, t_2)^T &= \mu_2\mathbf{X}^{-1}(1, \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}))^T \text{ and} \\ (z_1, z_2)^T &= \tau_2\mathbf{X}^T(\mathcal{H}_\ell(\mathbf{id}_k), -1)^T\end{aligned}$$

are uniformly and independently distributed. Note that all the information provided to the adversary except for the challenge ciphertext and the response to the k-th key extraction query are independent of the matrix $\mathbf{X}$. Also, observe that $\mathbf{id}_k \neq \widehat{\mathbf{id}}$ and the quantities $\theta_1, \ldots, \theta_h$ are uniformly distributed over $\mathbb{Z}_p$. Hence, by Lemma B.1 it follows that the coefficient vectors are uniformly and independently distributed except when $\mathcal{H}_\ell(\mathbf{id}_k) \equiv \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \pmod p$ which does not hold.

Observe that, if $\mathscr{B}$ tries to generate a semi-functional ciphertext for $\mathbf{id}_k$, the coefficient vectors of $\mathbf{b}_3^*, \mathbf{b}_4^*$ will end up being orthogonal which means that the ciphertext-key pair becomes nominally semi-functional. This provides no extra information to $\mathscr{B}$ about whether or not $\mathcal{SK}_{\mathsf{id}_k}$ is semi-functional or not.

It remains to bound $\mathsf{Adv}^{\text{DS2}}_{\mathcal{G}}(\mathscr{B})$. If $\beta = \beta'$, the $\mathscr{B}$ returns 1; otherwise it returns 0. Let Y_{real} and Y_{random} denote the events that $\mathscr{B}$ returns 1 when $\vec{T}_1$, $\vec{T}_2$ are real and random respectively; let abort denote the event that $\mathscr{B}$ aborts. We have

$$\begin{aligned}\Pr[Y_{real}] &= \Pr[Y_{real}|\mathsf{abort}]\Pr[\mathsf{abort}] + \Pr[Y_{real}|\overline{\mathsf{abort}}]\Pr[\overline{\mathsf{abort}}] \\ &= \frac{1}{2}\Pr[\mathsf{abort}] + \Pr[X_{k-1}]\Pr[\overline{\mathsf{abort}}], \qquad (3) \\ \Pr[Y_{random}] &= \Pr[Y_{random}|\mathsf{abort}]\Pr[\mathsf{abort}] + \Pr[Y_{random}|\overline{\mathsf{abort}}]\Pr[\overline{\mathsf{abort}}] \\ &= \frac{1}{2}\Pr[\mathsf{abort}] + \Pr[X_k]\Pr[\overline{\mathsf{abort}}] \qquad (4)\end{aligned}$$

Subtracting (3) from (4) gives us

$$\begin{aligned}\Pr[Y_{real}] - \Pr[Y_{random}] &= (\Pr[X_{k-1}] - \Pr[X_k])\Pr[\overline{\mathsf{abort}}] \\ &\ge (\Pr[X_{k-1}] - \Pr[X_k])\,\gamma_p^2\end{aligned}$$

and hence

$$\mathsf{Adv}^{\text{DS2}}_{\mathcal{V}_4}(\mathscr{B}) = |\Pr[Y_{real}] - \Pr[Y_{random}]| \ge \gamma_p^2\,|\Pr[X_{k-1}] - \Pr[X_k]|\,.$$

□

Lemma B.6. *Suppose an adversary $\mathscr{A}$ against $\mathcal{DPVS}$-$\mathcal{HIBE}$ makes at most q key extraction queries, then one can construct an algorithms $\mathscr{B}$ which has advantage $|\Pr[X_q] - \Pr[X_{final}]|$ in breaking the* DS1 *with $n = 4$ and $k = 2$.*

Proof. $\mathscr{B}$ receives an instance $(\mathcal{G}, \mathbf{d}_1 P_1, \ldots, \mathbf{d}_4 P_1, \mathbf{d}_1^* P_2, \mathbf{d}_2^* P_2, \vec{U}_1, \vec{U}_2, \mu_2, \vec{T}_1, \vec{T}_2)$ of DS1 problem and has to decide whether $\vec{T}_1, \vec{T}_2$ are distributed as $\tau_1 \mathbf{d}_1 P_1, \tau_1 \mathbf{d}_2 P_1$ or $(\tau_1 \mathbf{d}_1 + \tau_2 \mathbf{d}_3) P_1, (\tau_1 \mathbf{d}_2 + \tau_2 \mathbf{d}_4) P_1$ (whether they are real or random respectively) and accordingly return a bit 0 or 1. $\mathscr{B}$ will simulate each phase of the security game as described below.

Setup: $\mathscr{B}$ constructs the dual bases $\mathbf{B}, \mathbf{B}^*$ as follows.

$$(\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3, \mathbf{b}_4) = (\mathbf{d}_3, \mathbf{d}_4, \mathbf{d}_1, \mathbf{d}_2),$$

$$(\mathbf{b}_1^*, \mathbf{b}_2^*, \mathbf{b}_3^*, \mathbf{b}_4^*) = (\mathbf{d}_3^*, \mathbf{d}_4^*, \mathbf{d}_1^*, \mathbf{d}_2^*).$$

Note that $(\mathbf{B}, \mathbf{B}^*)$ are obtained by just a cyclic shift of the vectors in $(\mathbf{D}, \mathbf{D}^*)$. Then $\mathscr{B}$ chooses $\alpha', \theta_1, \ldots, \theta_h \xleftarrow{\mathrm{U}} \mathbb{Z}_p$, computes

$$\begin{aligned} \bar{e}(\mathbf{d}_3 P_1, \vec{U}_1)^{\alpha'} &= \bar{e}\left(\mathbf{d}_3 P_1, (\mu_1 \mathbf{d}_1^* + \mu_2 \mathbf{d}_3^*) P_2\right)^{\alpha'} \\ &= \bar{e}\left(\mathbf{d}_3 P_1, \mathbf{d}_3^* P_2\right)^{\mu_2 \alpha'} \\ &= e(P_1, P_2)^{\alpha \mathbf{b}_1 \cdot \mathbf{b}_1^*}, \end{aligned}$$

setting $\alpha = \alpha' \mu_2$, and provides the following public parameters to $\mathscr{A}$.

$$\mathcal{PP} = (\mathcal{G}, \mathbf{b}_1 P_1, \theta_1 \mathbf{b}_2 P_1, \ldots, \theta_h \mathbf{b}_2 P_1, e(P_1, P_2)^{\alpha \mathbf{b}_1 \cdot \mathbf{b}_1^*})$$

Note that $\alpha \mathbf{b}_1^* P_2$ is not known to $\mathscr{B}$ and hence it cannot create normal keys.

Key Extraction Phases 1 and 2: Let $\mathbf{id}_1, \ldots, \mathbf{id}_q$ be the identities provided by $\mathscr{A}$ for key extraction. The response to the k-th query $\mathbf{id}_k = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$, $\mathcal{SK}_{\mathbf{id}_k}$, is computed as follows. $\mathscr{B}$ picks $r', z_1', z_2', x_{\ell+1}', \ldots, x_h', y_{\ell+1}, \ldots, y_h \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ and computes

$$\vec{K} = (\alpha' + r' \mathcal{H}_\ell(\mathbf{id}_k)) \vec{U}_1 - r' \vec{U}_2 + z_1' \mathbf{d}_1^* P_2 + z_2' \mathbf{d}_2^* P_2,$$

$$\vec{D}_j = r' \theta_j \vec{U}_1 + x_j' \mathbf{d}_1^* P_2 + y_j \mathbf{d}_2^* P_2 \text{ for } \ell + 1 \le j \le h,$$

implicitly setting $r = r' \mu_2$, $z_1 = z_1' + \mu_1(\alpha' + r' \mathcal{H}_\ell(\mathbf{id}_k))$, $z_2 = z_2' - r' \mu_1$ and $x_j = x_j' + r' \theta_j \mu_1$ for $j = \ell + 1, \ldots, h$. The following computation shows that the key $\mathcal{SK}_{\mathbf{id}_k} = (\vec{K}, \vec{D}_{\ell+1}, \ldots, \vec{D}_h)$ is a properly distributed semi-functional key for $\mathbf{id}_k$.

$$\begin{aligned} \vec{K} &= (\alpha' + r' \mathcal{H}_\ell(\mathbf{id}_k)) \vec{U}_1 - r' \vec{U}_2 + z_1' \mathbf{d}_1^* P_2 + z_2' \mathbf{d}_2^* P_2 \\ &= (\alpha' + r' \mathcal{H}_\ell(\mathbf{id}_k))(\mu_1 \mathbf{d}_1^* + \mu_2 \mathbf{d}_3^*) P_2 - r'(\mu_1 \mathbf{d}_2^* + \mu_2 \mathbf{d}_4^*) P_2 + z_1' \mathbf{d}_1^* P_2 + z_2' \mathbf{d}_2^* P_2 \\ &= (\mu_2 \alpha' + \mu_2 r' \mathcal{H}_\ell(\mathbf{id}_k)) \mathbf{d}_3^* P_2 - r' \mu_2 \mathbf{d}_4^* P_2 + (z_1' + (\alpha' + r' \mathcal{H}_\ell(\mathbf{id}_k)) \mu_1) \mathbf{d}_1^* P_2 + (z_2' - r' \mu_1) \mathbf{d}_2^* P_2 \\ &= (\alpha + r \mathcal{H}_\ell(\mathbf{id}_k)) \mathbf{b}_1^* P_2 - r \mathbf{b}_2^* P_2 + z_1 \mathbf{b}_3^* P_2 + z_2 \mathbf{b}_4^* P_2 \end{aligned}$$

$$\begin{aligned} \vec{D}_j &= r' \theta_j \vec{U}_1 + x_j' \mathbf{d}_1^* P_2 + y_j \mathbf{d}_2^* P_2 \\ &= r' \theta_j (\mu_1 \mathbf{d}_1^* + \mu_2 \mathbf{d}_3^*) P_2 + x_j' \mathbf{d}_1^* P_2 + y_j \mathbf{d}_2^* P_2 \\ &= r' \theta_j \mu_2 \mathbf{d}_3^* P_2 + (x_j' + r' \theta_j \mu_1) \mathbf{d}_1^* P_2 + y_j \mathbf{d}_2^* P_2 \\ &= r \theta_j \mathbf{b}_1^* P_2 + x_j \mathbf{b}_3^* P_2 + y_j \mathbf{b}_4^* P_2 \end{aligned}$$

Challenge: $\mathscr{A}$ sends two messages M_0, M_1 and the challenge identity vector $\widehat{\mathbf{id}} = (\widehat{\mathsf{id}}_1, \ldots, \widehat{\mathsf{id}}_{\widehat{\ell}})$. The simulator now chooses $\beta \xleftarrow{\text{U}} \{0,1\}$, $s', t_1', t_2' \xleftarrow{\text{U}} \mathbb{Z}_p$ and computes the challenge ciphertext as $\widehat{\mathcal{C}} = (C_0, \vec{C}_1)$ as follows.

$$C_0 = M_\beta \cdot \bar{e}(\mathbf{d}_3 P_1, \vec{U}_1)^{\alpha' s'},$$
$$\vec{C}_1 = s'\mathbf{d}_3 P_1 + s'\mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})\mathbf{d}_4 P_1 + \vec{T}_1 + \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})\vec{T}_2 + t_1'\mathbf{d}_1 + t_2'\mathbf{d}_2.$$

If $\vec{T}_1$,$\vec{T}_2$ are real then

$$\begin{aligned}\vec{C}_1 &= s'\mathbf{d}_3 P_1 + s'\mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})\mathbf{d}_4 P_1 + \tau_1\mathbf{d}_1 P_1 + \tau_1\mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})\mathbf{d}_2 P_1 + t_1'\mathbf{d}_1 + t_2'\mathbf{d}_2 \\ &= s'\mathbf{b}_1 P_1 + s'\mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})\mathbf{b}_2 P_1 + (t_1' + \tau_1)\mathbf{b}_3 P_1 + (t_2' + \tau_1\mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}))\mathbf{b}_4 P_1,\end{aligned}$$

making $\widehat{\mathcal{C}}$ a semi-functional encryption of M_β with $s = s'$, $t_1 = t_1' + \tau_1$ and $t_2 = t_2' + \tau_1\mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})$. Thus $\mathscr{B}$ simulates Game_q. Otherwise, if $\vec{T}_1$,$\vec{T}_2$ are random, $\widehat{\mathcal{C}}$ will be a semi-functional encryption of a random message and as a result $\mathscr{B}$ simulates Game_{final}. This is because C_1 is given by

$$\begin{aligned}\vec{C}_1 &= (s' + \tau_2)\mathbf{d}_3 P_1 + (s' + \tau_2)\mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})\mathbf{d}_4 P_1 + (t_1' + \tau_1)\mathbf{d}_1 P_1 + (t_2' + \tau_1\mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}))\mathbf{d}_2 P_1 \\ &= (s' + \tau_2)\mathbf{b}_1 P_1 + (s' + \tau_2)\mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})\mathbf{b}_2 P_1 + (t_1' + \tau_1)\mathbf{b}_3 P_1 + (t_2' + \tau_1\mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}))\mathbf{b}_4 P_1\end{aligned}$$

with $s = s' + \tau_2$ which in turn causes the message M_β to be randomised by the quantity $e(P_1, P_2)^{-\alpha\tau_2 \mathbf{b}_1 \cdot \mathbf{b}_1^*}$ in C_0.

$\mathscr{B}$ returns 1 if $\mathscr{A}$ wins the game and 0 otherwise. Let Y_{real} and Y_{random} denote the events that $\mathscr{B}$ returns 1 in Game_q and Game_{final} respectively. Then,

$$\mathsf{Adv}^{\text{DS1}}_{\mathcal{V}_4}(\mathscr{B}) = |\Pr[Y_{real}] - \Pr[Y_{random}]| = |\Pr[X_q] - \Pr[X_{final}]|$$

□

Theorem B.7. *If the* $(\varepsilon_{\text{DDH1}}, t_{\text{DDH1}})$-DDH1 *and* $(\varepsilon_{\text{DDH2}}, t_{\text{DDH2}})$-DDH2 *assumptions hold in* $\mathcal{G}$*, then* $\mathcal{DPVS}$*-*$\mathcal{HIBE}$ *is* (ε, t, q)-IND-ID-CPA *secure where*

$$\varepsilon \leq \left(1 + \frac{1}{\gamma_p^2}\right)\varepsilon_{\text{DDH1}} + \frac{q}{\gamma_p^2}\varepsilon_{\text{DDH2}} + \frac{q}{p}$$

and $t_{\text{DDH1}} = t + O(h\rho)$, $t_{\text{DDH2}} = t + O(qh\rho)$ *where* ρ *is the maximum time for one scalar multiplication in either of* $\mathbb{G}_1$,$\mathbb{G}_2$.

Proof. Using lemmas B.3, B.4, B.5 and B.6, we have for any t-time adversary $\mathscr{A}$ against $\mathcal{DPVS}$*-*$\mathcal{HIBE}$ in the ind-cpa game, there is an algorithm $\mathscr{B}$ such that

$$\begin{aligned}\mathsf{Adv}^{\mathsf{ind-cpa}}_{\mathcal{DPVS}\text{-}\mathcal{HIBE}}(\mathscr{A}) &= \left|\Pr[X_{actual}] - \frac{1}{2}\right| \\ &= |\Pr[X_{actual}] - \Pr[X_{final}]| \\ &\leq |\Pr[X_{actual}] - \Pr[X_{restricted}]| + |\Pr[X_{restricted}] - \Pr[X_0]| \\ &\quad + \sum_{k=1}^{q}(|\Pr[X_{k-1}] - \Pr[X_k]|) + |\Pr[X_q] - \Pr[X_{final}]| \\ &\leq \frac{q}{p} + \frac{1}{\gamma_p^2}\mathsf{Adv}^{\text{DS1}}_{\mathcal{V}_4}(\mathscr{B}) + q\left(\frac{1}{\gamma_p^2}\mathsf{Adv}^{\text{DS2}}_{\mathcal{V}_4}(\mathscr{B})\right) + \mathsf{Adv}^{\text{DS1}}_{\mathcal{V}_4}(\mathscr{B}) \\ &= \left(1 + \frac{1}{\gamma_p^2}\right)\mathsf{Adv}^{\text{DS1}}_{\mathcal{G}}(\mathscr{B}) + \frac{q}{\gamma_p^2}\mathsf{Adv}^{\text{DS2}}_{\mathcal{G}}(\mathscr{B}) + \frac{q}{p}.\end{aligned}$$

The theorem now follows directly from Lemma A.1 since n and k are constant in the DS1, DS2 problems we consider. □

C Security Proof for *DPVS-AHIBE*

Semi-functional ciphertexts for *DPVS-AHIBE* are defined in the same way as for *DPVS-HIBE* in Section B. We only define semi-functional keys and also the notion of partial semi-functional key in order to explain why we need partial semi-functionality. Also, these would be used in the proof.

Semi-functional key: Let $(\vec{K}_1, \vec{D}_{1,\ell+1}, \ldots, \vec{D}_{1,h}, \vec{K}_2, \vec{D}_{2,\ell+1}, \ldots, \vec{D}_{2,h})$ be a normal key for an identity vector $\mathbf{id}$ of length ℓ. A semi-functional key for $\mathbf{id}$ is computed as follows:

$$z_1, z_2, w_1, w_2, (x_{1,j}), (y_{1,j}), (x_{2,j}), (y_{2,j}) \xleftarrow{\text{U}} \mathbb{Z}_p \text{ for } \ell+1 \le j \le h,$$

$$\vec{K}_1 \leftarrow \vec{K}_1 + (z_1\mathbf{b}_3^* + z_2\mathbf{b}_4^*)P_2,\ \vec{D}_{1,j} \leftarrow \vec{D}_{1,j} + x_{1,j}\mathbf{b}_3^*P_2 + y_{1,j}\mathbf{b}_4^*P_2 \quad \text{for } \ell+1 \le j \le h$$

$$\vec{K}_2 \leftarrow \vec{K}_2 + (w_1\mathbf{b}_3^* + w_2\mathbf{b}_4^*)P_2,\ \vec{D}_{2,j} \leftarrow \vec{D}_{2,j} + x_{2,j}\mathbf{b}_3^*P_2 + y_{2,j}\mathbf{b}_4^*P_2 \quad \text{for } \ell+1 \le j \le h$$

Partial semi-functional key: In a partial semi-functional key the components $\vec{K}_2, (\vec{D}_{2,j})$ are normal and the other vectors have semi-functional components as defined in the previous paragraph.

Note: In the ano-ind-cpa game, CPA-security is captured based on the model proposed by Gentry-Silverberg [GS02] which requires that the distribution of keys output by the KeyGen algorithm is the same as the distribution of keys obtained by running the Delegate algorithm. Hence it becomes important for our security argument that the distribution of the keys remains the same after delegation irrespective of whether they are normal or (partial) semi-functional. Indeed, this holds for our scheme by construction and the definitions of (partial) semi-functional keys.

We now present the proof of security for *DPVS-AHIBE*. The structure of the proof is different here. The sequence of games over which we make the hybrid argument now consists of $2q+4$ games as defined below Again, the shorthand $\mathcal{H}_\ell(\mathbf{id})$ will be used to denote the quantity $\sum_{j=1}^{\ell} \mathsf{id}_j\theta_j$ where $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$ and $\gamma_p = \left(1 - \frac{1}{p}\right)$.

Game$_{actual}$: The ano-ind-cpa game defined in Section A.2.2.

Game$_{restricted}$: Similar to Game$_{actual}$ but with the added restriction that $\mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \not\equiv \mathcal{H}_{\ell_k}(\mathbf{id}_k) \pmod p$ for all $k \in \{1, \ldots, q\}$.

Game$_0$: In this game, the challenge ciphertext is semi-functional and all the keys returned to the adversary are normal.

Game$_{k,0}$ (for $1 \le k \le q$): In this game, k-th key is partial semi-functional, the first $k-1$ keys are semi-functional; the rest of the keys are normal.

Game$_{k,1}$ (for $1 \le k \le q$): Similar to Game$_{k,0}$ except that the k-th key is (fully) semi-functional.

Game$_{final}$: All keys are semi-functional and the challenge ciphertext is a semi-functional encryption of a random message to a random identity vector.

These games are ordered as $\text{Game}_{actual}, \text{Game}_{restricted}, \text{Game}_0, \text{Game}_{1,0}, \text{Game}_{1,1}, \ldots, \text{Game}_{q,0}, \text{Game}_{q,1}, \text{Game}_{final}$ in our hybrid argument. Let X_{actual}, $\text{Game}_{restricted}$, $X_{k,0}$, $X_{k,1}$ and X_{final} denote events that the adversary wins in Game_{actual}, $\text{Game}_{restricted}$, $\text{Game}_{k,0}$, $\text{Game}_{k,1}$ and Game_{final} for $1 \leq k \leq q$ respectively. Also, $\text{Game}_{0,1}$ is the same as Game_0.

Lemma C.1. $|\Pr[X_{actual}] - \Pr[X_{restricted}]| \leq q/p$.

Lemma C.2. *If there exists a adversary $\mathscr{A}$ against DPVS-AHIBE making at most q key extraction queries, then we can build an algorithm $\mathscr{B}$ having advantage at least $\gamma_p^2 |\Pr[X_{restricted}] - \Pr[X_0]|$ in solving the* DS1 *problem with $n = 4, k = 2$.*

The proof of this lemma is similar to that of Lemma B.4 and hence we omit the proof.

Lemma C.3. *If there exists an adversary $\mathscr{A}$ against DPVS-AHIBE making at most q key extraction queries then we can build an algorithm $\mathscr{B}$ having advantage at least $\gamma_p^2 |\Pr[X_{k-1,1}] - \Pr[X_{k,0}]|$ in solving the* DS2 *problem with $n = 4, k = 2$.*

Proof. An instance $(\mathcal{G}, \mathbf{d}_1 P_1, \mathbf{d}_2 P_1, \mathbf{d}_1^* P_2, \ldots, \mathbf{d}_4^* P_2, \vec{U}_1, \vec{U}_2, \mu_2, \vec{T}_1, \vec{T}_2)$ of DS2 is provided to $\mathscr{B}$. Here $\vec{U}_1 = (\mu_1 \mathbf{d}_1 + \mu_2 \mathbf{d}_3) P_1$ and $\vec{U}_2 = (\mu_1 \mathbf{d}_2 + \mu_2 \mathbf{d}_4) P_1$. $\mathscr{B}$ has to determine whether $\vec{T}_1, \vec{T}_2$ are real or random - i.e., whether they are distributed as $\tau_1 \mathbf{d}_1^* P_2, \tau_1 \mathbf{d}_2^* P_2$ or $(\tau_1 \mathbf{d}_1^* + \tau_2 \mathbf{d}_3^*) P_2, (\tau_1 \mathbf{d}_2^* + \tau_2 \mathbf{d}_4^*) P_2$.

Setup: $\mathscr{B}$ samples $\mathbf{X} = (\chi_{i,j}) \xleftarrow{\text{U}} \mathbb{Z}_p^{2\times 2}$. If $\mathbf{X}$ is not invertible, it aborts and returns a random bit; otherwise it proceeds to set the dual bases $(\mathbf{B}, \mathbf{B}^*)$ of the HIBE scheme as follows.

$$\mathbf{b}_1 = \mathbf{d}_1, \;\; \mathbf{b}_2 = \mathbf{d}_2, \;\; (\mathbf{b}_3, \mathbf{b}_4) = (\mathbf{d}_3, \mathbf{d}_4)\mathbf{X},$$
$$\mathbf{b}_1^* = \mathbf{d}_1^*, \;\; \mathbf{b}_2^* = \mathbf{d}_2^*, \;\; (\mathbf{b}_3^*, \mathbf{b}_4^*) = (\mathbf{d}_3^*, \mathbf{d}_4^*)(\mathbf{X}^{-1})^T.$$

$\mathscr{B}$ then picks $\alpha, \theta_1, \ldots, \theta_h \xleftarrow{\text{U}} \mathbb{Z}_p$ and sends the public parameters $\mathcal{PP}$ to $\mathscr{A}$. $\mathscr{B}$ knows $\mathcal{MSK}$ and the vectors $\mathbf{b}_3^*, \mathbf{b}_4^*$ and so it should be able to create normal as well as semi-functional keys.

Phases 1 and 2: Let $\mathbf{id}_1, \mathbf{id}_2, \ldots, \mathbf{id}_q$ denote the identities for which $\mathscr{A}$ requests the secret keys. For $i > k$, $\mathscr{B}$ generates a normal key for $\mathbf{id}_i$ using the master secret and returns the resulting key to $\mathscr{A}$. For $i < k$, $\mathscr{B}$ first generates a normal key and then modifies it appropriately to obtain a semi-functional key. The resulting key is returned to $\mathscr{A}$. In case of the k-th query ($\mathbf{id}_k = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$), $\mathscr{B}$ generates the key as follows. Choose $r_2 \xleftarrow{\text{U}} \mathbb{Z}_p$ and set

$$\vec{K}_1 = \alpha \mathbf{b}_1^* P_2 + \mathcal{H}_\ell(\mathbf{id}_k)\vec{T}_1 - \vec{T}_2,$$
$$\vec{D}_{1,\ell+1} = \theta_{\ell+1}\vec{T}_1, \;\; \ldots, \;\; \vec{D}_{1,h} = \theta_h \vec{T}_1,$$
$$\vec{K}_2 = r_2 \mathcal{H}_\ell(\mathbf{id}_k)\mathbf{b}_1^* P_2 - r_2 \mathbf{b}_2^* P_2,$$
$$\vec{D}_{2,\ell+1} = r_2\theta_{\ell+1}\mathbf{b}_1^* P_2, \;\; \ldots, \;\; \vec{D}_{2,h} = r_2\theta_h \mathbf{b}_2^* P_2.$$

The key $\mathcal{SK}_{\mathbf{id}_k} = (\vec{K}_1, \vec{D}_{1,\ell+1}, \ldots, \vec{D}_{1,h}, \vec{K}_2, \vec{D}_{2,\ell+1}, \ldots, \vec{D}_{2,h})$ is returned to $\mathscr{A}$.

If $\vec{T}_1, \vec{T}_2$ are real, it is easy to see that $\mathcal{SK}_{\mathbf{id}_k}$ will be normal with $r_1 = \tau_1$ whence $\mathscr{B}$ simulates $\text{Game}_{k-1,1}$. Otherwise if $\vec{T}_1, \vec{T}_2$ are random, the key will be partial semi-functional as justified below.

$$\begin{aligned}
\vec{K}_1 &= \alpha \mathbf{b}_1^* P_2 + \mathcal{H}_\ell(\mathbf{id}_k)\vec{T}_1 - \vec{T}_2 \\
&= \alpha \mathbf{b}_1^* P_2 + \tau_1 \mathcal{H}_\ell(\mathbf{id}_k)\mathbf{b}_1^* P_2 - \tau_1 \mathbf{b}_2^* P_2 + (\mathbf{b}_3^*, \mathbf{b}_4^*)\mathbf{X}^T \tau_2 (\mathcal{H}_\ell(\mathbf{id}_k), -1)^T P_2, \\
\vec{D}_{1,j} &= \theta_j \vec{T}_1 \\
&= \theta_j \tau_1 \mathbf{b}_1^* P_2 + \theta_j \tau_2 (\chi_{1,1}\mathbf{b}_3^* + \chi_{1,2}\mathbf{b}_4^*) P_2,
\end{aligned}$$

setting $r_1 = \tau_1$, $(z_1, z_2)^T = \tau_2 \mathbf{X}^T (\mathcal{H}_\ell(\mathbf{id}_k), -1)^T$, $x_{1,j} = \theta_j \tau_2 \chi_{1,1}$ and $y_{1,j} = \theta_j \tau_2 \chi_{1,2}$ for $\ell + 1 \leq j \leq h$. In this case, $\mathscr{B}$ simulates $\text{Game}_{k,0}$.

Challenge: $\mathscr{B}$ receives two message-identity pairs $(M_0, \widehat{\mathbf{id}}_0)$, $(M_1, \widehat{\mathbf{id}}_1)$ from $\mathscr{A}$. It chooses $\beta \in \{0, 1\}$ at random and encrypts M_β to $\widehat{\mathbf{id}}_\beta = (\widehat{\mathsf{id}}_1, \ldots, \widehat{\mathsf{id}}_{\widehat{\ell}})$ as

$$C_0 = M_\beta \cdot \bar{e}(\vec{U}_1, \mathbf{b}_1^* P_2)^\alpha, \;\; \vec{C}_1 = \vec{U}_1 + \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}_\beta) \vec{U}_2$$

and returns $\widehat{\mathcal{C}} = (C_0, \vec{C}_1)$ to $\mathscr{A}$. Clearly, the vector $\vec{C}_1$ is properly distributed with $s = \mu_1$ and $(t_1, t_2)^T = \mu_2 \mathbf{X}^{-1} (1, \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}_\beta))^T$.

Guess: $\mathscr{A}$ returns its guess β' of β.

In $\text{Game}_{k-1,1}$, all information provided to the adversary is properly distributed. Due to the choice of the θ_j's and $\mathbf{X}$, in $\text{Game}_{k,0}$, the scalars $x_{1,j}$, $y_{1,j}$ (for $\ell + 1 \leq j \leq h$) will be uniformly distributed in $\mathbb{Z}_p$. Consequently, the semi-functional components of $\vec{D}_{1,j}$'s and $\vec{D}_{2,j}$'s will be properly distributed. We require that (in $\text{Game}_{k,0}$) the semi-functional components of $\mathcal{SK}_{\mathbf{id}_k}$ and $\widehat{\mathcal{C}}$ are independently distributed in $\mathscr{A}$'s view. This holds since Lemma B.1 implies that $(t_1, t_2)^T, (z_1, z_2)^T \in_{\text{U}} \mathbb{Z}_p^2$ unless the relation $\mathcal{H}_\ell(\mathbf{id}_k) \equiv \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}_\beta) \pmod p$ that does not occur.

We can now proceed to obtain a bound on $\mathsf{Adv}_{\mathcal{V}_4}^{\text{DS2}}(\mathscr{B})$. If $\beta = \beta'$, the $\mathscr{B}$ returns 1; otherwise it returns 0. Let Y_{real} and Y_{random} denote the events that $\mathscr{B}$ returns 1 when $\vec{T}_1, \vec{T}_2$ are real and random respectively; let abort denote the event that $\mathscr{B}$ aborts. We have

$$\begin{aligned}
\Pr[Y_{real}] &= \Pr[Y_{real} | \mathsf{abort}] \Pr[\mathsf{abort}] + \Pr[Y_{real} | \overline{\mathsf{abort}}] \Pr[\overline{\mathsf{abort}}] \\
&= \frac{1}{2} \Pr[\mathsf{abort}] + \Pr[X_{k-1,1}] \Pr[\overline{\mathsf{abort}}], \qquad (5) \\
\Pr[Y_{random}] &= \Pr[Y_{random} | \mathsf{abort}] \Pr[\mathsf{abort}] + \Pr[Y_{random} | \overline{\mathsf{abort}}] \Pr[\overline{\mathsf{abort}}] \\
&= \frac{1}{2} \Pr[\mathsf{abort}] + \Pr[X_{k,0}] \Pr[\overline{\mathsf{abort}}] \qquad (6)
\end{aligned}$$

Subtracting (5) from (6) gives us

$$\begin{aligned}
\Pr[Y_{real}] - \Pr[Y_{random}] &= (\Pr[X_{k-1,1}] - \Pr[X_{k,0}] -) \Pr[\overline{\mathsf{abort}}] \\
&\geq (\Pr[X_{k-1,1}] - \Pr[X_{k,0}]) \, \gamma_p^2
\end{aligned}$$

and hence

$$\begin{aligned}
\mathsf{Adv}_{\mathcal{V}_4}^{\text{DS2}}(\mathscr{B}) &= |\Pr[Y_{real}] - \Pr[Y_{random}]| \\
&\geq \gamma_p^2 \, |\Pr[X_{k-1,1}] - \Pr[X_{k,0}]| \, .
\end{aligned}$$

□

Lemma C.4. *If there exists an adversary $\mathscr{A}$ against $\mathcal{DPVS}$-$\mathcal{AHIBE}$ making at most q key extraction queries such that , then we can build an algorithm $\mathscr{B}$ having advantage at least $\gamma_p^2 \, |\Pr[X_{k,0}] - \Pr[X_{k,1}]|$ in solving the* DS2 *problem with $n = 4, k = 2$.*

Proof. $\mathscr{B}$ receives a DS2 instance $(\mathcal{G}, \mathbf{d}_1 P_1, \mathbf{d}_2 P_1, \mathbf{d}_1^* P_2, \ldots, \mathbf{d}_4^* P_2, \vec{U}_1, \vec{U}_2, \mu_2, \vec{T}_1, \vec{T}_2)$ with $\vec{U}_1 = (\mu_1 \mathbf{d}_1 + \mu_2 \mathbf{d}_3) P_1$ and $\vec{U}_2 = (\mu_1 \mathbf{d}_2 + \mu_2 \mathbf{d}_4) P_1$. $\mathscr{B}$ has to determine whether $\vec{T}_1, \vec{T}_2$ are real or random - i.e., whether they are distributed as $\tau_1 \mathbf{d}_1^* P_2, \tau_1 \mathbf{d}_2^* P_2$ or $(\tau_1 \mathbf{d}_1^* + \tau_2 \mathbf{d}_3^*) P_2, (\tau_1 \mathbf{d}_2^* + \tau_2 \mathbf{d}_4^*) P_2$.

The proof goes more or less similar to that of Lemma C.3. We only highlight the differences here. The setup phase is exactly the same. In the the key extraction phases, the secret key for the k-th identity vector, $\mathbf{id}_k = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$, is generated as follows. Choose $r_1, z_1, z_2, (x_{1,j}), (y_{1,j}) \xleftarrow{\text{U}} \mathbb{Z}_p$ and compute

$$\begin{aligned}
\vec{K}_1 &= \alpha \mathbf{b}_1^* P_2 + r_1 \mathcal{H}_\ell(\mathbf{id}_k) \mathbf{b}_1^* P_2 - r_1 \mathbf{b}_2^* P_2 + z_1 \mathbf{b}_3^* P_2, + z_2 \mathbf{b}_4^* P_2, \\
\vec{D}_{1,j} &= r_1 \theta_j \mathbf{b}_1^* P_2 + x_{1,j} \mathbf{b}_3^* P_2 + y_{1,j} \mathbf{b}_4^* P_2, \\
\vec{K}_2 &= \mathcal{H}_\ell(\mathbf{id}_k) \vec{T}_1 - \vec{T}_2, \\
\vec{D}_{2,j} &= \theta_j \vec{T}_1,
\end{aligned}$$

where $\ell + 1 \leq j \leq h$. The key $\mathcal{SK}_{\mathbf{id}_k} = (\vec{K}_1, \vec{D}_{1,\ell+1}, \ldots, \vec{D}_{1,h}, \vec{K}_2, \vec{D}_{2,\ell+1}, \ldots, \vec{D}_{2,h})$ is returned to $\mathscr{A}$.

If $\vec{T}_1, \vec{T}_2$ are real, it is easy to see that $\mathcal{SK}_{\mathbf{id}_k}$ will be partial semi-functional with $r_1, z_1, z_2, (x_{1,j}), (y_{1,j})$ being the scalars defining the semi-functional components of the first part of the key (i.e., for $\vec{K}_2, (\vec{D}_{2,j})$) and the randomiser for the second part given by $r_2 = \tau_2$. Hence $\mathscr{B}$ simulates $\text{Game}_{k,0}$. Otherwise if $\vec{T}_1, \vec{T}_2$ are random, the key will be semi-functional as illustrated by the following computation.

$$\begin{aligned}
\vec{K}_2 &= \mathcal{H}_\ell(\mathbf{id}_k) \vec{T}_1 - \vec{T}_2 \\
&= \tau_1 \mathcal{H}_\ell(\mathbf{id}_k) \mathbf{b}_1^* P_2 - \tau_1 \mathbf{b}_2^* P_2 + (\mathbf{b}_3^*, \mathbf{b}_4^*) \mathbf{X}^T \tau_2 (\mathcal{H}_\ell(\mathbf{id}_k), -1)^T P_2, \\
\vec{D}_{2,j} &= \theta_j \vec{T}_1 \\
&= \theta_j \tau_1 \mathbf{b}_1^* P_2 + \theta_j \tau_2 (\chi_{1,1} \mathbf{b}_3^* + \chi_{1,2} \mathbf{b}_4^*) P_2,
\end{aligned}$$

setting $r_2 = \tau_1$, $(w_1, w_2)^T = \tau_2 \mathbf{X}^T (\mathcal{H}_\ell(\mathbf{id}_k), -1)^T$, $x_{2,j} = \theta_j \tau_2 \chi_{1,1}$ and $y_{2,j} = \theta_j \tau_2 \chi_{1,2}$ for $\ell + 1 \leq j \leq h$. In this case, $\mathscr{B}$ simulates $\text{Game}_{k,1}$.

The challenge phase is again identical to the one given in Lemma C.3. The randomisers for the ciphertext are given by $s = \mu_1$ and $(t_1, t_2)^T = \mu_2 \mathbf{X}^{-1} (1, \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}_\beta))^T$.

In $\text{Game}_{k,0}$, all information provided to the adversary is properly distributed by the choice of the scalars $z_1, z_2, (x_{1,j}), (y_{1,j})$ and r_1. In $\text{Game}_{k,1}$, because of the way θ_j's and $\mathbf{X}$ are chosen, the scalars $x_{2,j}$, $y_{2,j}$ (for $\ell + 1 \leq j \leq h$) will be uniformly and independently distributed $\mathbb{Z}_p$. Also these scalars will be independent of the choice of $z_1, z_2, (x_{1,j}), (y_{1,j})$ and r_1. As a result, the semi-functional components of $\vec{K}_2$ and $\vec{D}_{2,j}$'s will have the proper distribution. By a similar argument using Lemma B.1, we conclude that $(t_1, t_2)^T, (w_1, w_2)^T \in_{\text{U}} \mathbb{Z}_p^2$, unless $\mathcal{H}_\ell(\mathbf{id}_k) \equiv \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}_\beta) \pmod p$.

The following bound on the advantage of $\mathscr{B}$ in solving the DS2 problem is obtained in a similar fashion and we omit the details here.

$$\mathsf{Adv}^{\text{DS2}}_{\mathcal{V}_4}(\mathscr{B}) \geq \gamma_p^2 \left(|\Pr[X_{k,0}] - \Pr[X_{k,1}]| \right).$$

□

Lemma C.5. *For any PPT adversary $\mathscr{A}$ against DPVS-AHIBE, $\Pr[X_{q,1}] = \Pr[X_{final}]$.*

Proof. Let $\mathcal{E}_{q,1}$ and $\mathcal{E}_{final}$ denote the distribution of all the information provided to the adversary in $\text{Game}_{q,1}$ and Game_{final} respectively.

$$\mathcal{E}_{q,1} = (\mathcal{PP}, \widehat{\mathcal{C}}, (\mathcal{SK}_{\mathbf{id}_k})_{k=1,\ldots,q})$$

where $\widehat{\mathcal{C}}$ is a semi-functional encryption of M_β to $\widehat{\mathbf{id}}_\beta$ and $\mathcal{SK}_{\mathbf{id}_k}$ is a semi-functional key for $\mathbf{id}_k$.

$$\mathcal{E}_{final} = (\mathcal{PP}, \mathcal{C}^{(R)}, (\mathcal{SK}_{\mathbf{id}_k})_{k=1,\ldots,q})$$

where $\mathcal{C}^{(R)}$ is an encryption of a random message to a random identity vector. The goal here is to show that $\mathcal{E}_{q,1}$ and $\mathcal{E}_{final}$ are statistically indistinguishable. This is done by applying a change of basis transformation to the dual bases $\mathbf{B}, \mathbf{B}^*$ keeping the public parameters intact and then showing the view of the adversary under the new bases is statistically indistinguishable from the view before the transformation. Details follow.

Pick $\mathbf{X} = (\chi_{\imath,\jmath}) \xleftarrow{\mathrm{U}} \mathbb{Z}_p^{2\times 2}$ and define new dual bases $\mathbf{F} = (\mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3, \mathbf{f}_4)$ and $\mathbf{F}^* = (\mathbf{f}_1^*, \mathbf{f}_2^*, \mathbf{f}_3^*, \mathbf{f}_4^*)$ as:

$$\left(\begin{array}{cccc} \mathbf{f}_1 & \mathbf{f}_2 & \mathbf{f}_3 & \mathbf{f}_4 \end{array} \right) = \left(\begin{array}{cccc} \mathbf{b}_1 & \mathbf{b}_2 & \mathbf{b}_3 & \mathbf{b}_4 \end{array} \right) \begin{pmatrix} 1 & 0 & \chi_{1,1} & \chi_{2,1} \\ 0 & 1 & \chi_{1,2} & \chi_{2,2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$\left(\begin{array}{cccc} \mathbf{f}_1^* & \mathbf{f}_2^* & \mathbf{f}_3^* & \mathbf{f}_4^* \end{array} \right) = \left(\begin{array}{cccc} \mathbf{b}_1^* & \mathbf{b}_2^* & \mathbf{b}_3^* & \mathbf{b}_4^* \end{array} \right) \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -\chi_{1,1} & -\chi_{1,2} & 1 & 0 \\ -\chi_{2,1} & -\chi_{2,2} & 0 & 1 \end{pmatrix}$$

Clearly $(\mathbf{F}, \mathbf{F}^*)$ are dual bases with the same distribution as $(\mathbf{B}, \mathbf{B}^*)$.

Information provided to the adversary (i.e., public parameters, challenge ciphertext and responses to key extract queries) in $\mathrm{Game}_{q,1}$ when expressed over bases $(\mathbf{B}, \mathbf{B}^*)$ will be

$$\mathcal{PP} = (\mathcal{G}, \mathbf{b}_1 P_1, \theta_1 \mathbf{b}_2 P_2, \ldots, \theta_h \mathbf{b}_2 P_1, e(P_1, P_2)^{\alpha \mathbf{b}_1 \cdot \mathbf{b}_1^*}),$$

$$\widehat{\mathcal{C}} = \left(C_0 = M_\beta \cdot e(P_1, P_2)^{s\alpha \mathbf{b}_1 \cdot \mathbf{b}_1^*}, \vec{C}_1 = (s\mathbf{b}_1 + s\mathcal{H}_{\hat{\ell}}(\widehat{\mathbf{id}}_\beta)\mathbf{b}_2 + t_1\mathbf{b}_3 + t_2\mathbf{b}_4)P_1 \right),$$

$$\mathcal{SK}_{\mathbf{id}_k} = \left(\vec{K}_1^{(k)}, \vec{D}_{1,\ell^{(k)}+1}^{(k)}, \ldots, \vec{D}_{1,h}^{(k)}, \vec{D}_{2,\ell^{(k)}+1}^{(k)}, \ldots, \vec{D}_{2,h}^{(k)} \right) \text{ for } k = 1, \ldots, q$$

where

$$\vec{K}_1^{(k)} = \left(\alpha + r_1^{(k)} \mathcal{H}_{\ell^{(k)}}(\mathbf{id}_k)\mathbf{b}_1^* - r_1^{(k)}\mathbf{b}_2^* + z_1^{(k)}\mathbf{b}_3^* + z_2^{(k)}\mathbf{b}_4^* \right) P_2,$$

$$\vec{D}_{1,j}^{(k)} = \left(r_1\theta_j \mathbf{b}_1^* + x_{1,j}^{(k)}\mathbf{b}_3^* + y_{1,j}^{(k)}\mathbf{b}_4^* \right) P_2 \text{ for } \ell^{(k)} + 1 \le j \le h,$$

$$\vec{K}_2^{(k)} = \left(r_2^{(k)} \mathcal{H}_{\ell^{(k)}}(\mathbf{id}_k)\mathbf{b}_1^* - r_2^{(k)}\mathbf{b}_2^* + w_1^{(k)}\mathbf{b}_3^* + w_2^{(k)}\mathbf{b}_4^* \right) P_2,$$

$$\vec{D}_{2,j}^{(k)} = \left(r_2\theta_j \mathbf{b}_1^* + x_{2,j}^{(k)}\mathbf{b}_3^* + y_{2,j}^{(k)}\mathbf{b}_4^* \right) P_2 \text{ for } \ell^{(k)} + 1 \le j \le h.$$

These can be expressed over bases $(\mathbf{F}, \mathbf{F}^*)$ as

$$\mathcal{PP} = (\mathcal{G}, \mathbf{f}_1 P_1, \theta_1 \mathbf{f}_2 P_2, \ldots, \theta_h \mathbf{f}_2 P_1, e(P_1, P_2)^{\alpha \mathbf{f}_1 \cdot \mathbf{f}_1^*}),$$

$$\widehat{\mathcal{C}} = \left(C_0 = M_\beta \cdot e(P_1, P_2)^{s\alpha \mathbf{f}_1 \cdot \mathbf{f}_1^*}, \vec{C}_1 = (s'\mathbf{f}_1 + s''\mathbf{f}_2 + t_1\mathbf{f}_3 + t_2\mathbf{f}_4)P_1 \right)$$

and for $k = 1, \ldots, q$,

$$\vec{K}_1^{(k)} = \left(\alpha + r_1^{(k)} \mathcal{H}_{\ell^{(k)}}(\mathbf{id}_k)\mathbf{f}_1^* - r_1^{(k)}\mathbf{f}_2^* + (z_1')^{(k)}\mathbf{f}_3^* + (z_2')^{(k)}\mathbf{f}_4^* \right) P_2,$$

$$\vec{D}_{1,j}^{(k)} = \left(r_1\theta_j \mathbf{f}_1^* + (x_{1,j}')^{(k)}\mathbf{f}_3^* + (y_{1,j}')^{(k)}\mathbf{f}_4^* \right) P_2 \text{ for } \ell^{(k)} + 1 \le j \le h,$$

$$\vec{K}_2^{(k)} = \left(r_2^{(k)} \mathcal{H}_{\ell^{(k)}}(\mathbf{id}_k)\mathbf{f}_1^* - r_2^{(k)}\mathbf{f}_2^* + (w_1')^{(k)}\mathbf{f}_3^* + (w_2')^{(k)}\mathbf{f}_4^* \right) P_2,$$

$$\vec{D}_{2,j}^{(k)} = \left(r_2\theta_j \mathbf{f}_1^* + (x'_{2,j})^{(k)}\mathbf{f}_3^* + (y'_{2,j})^{(k)}\mathbf{f}_4^*\right) P_2 \text{ for } \ell^{(k)}+1 \le j \le h$$

where

$$s' = s - t_1\chi_{1,1} - t_2\chi_{2,1},$$
$$s'' = s\mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}_\beta) - t_1\chi_{1,2} - t_2\chi_{2,2}$$

and for $k = 1, \ldots, q$,

$$(z'_1)^{(k)} = z_1^{(k)} + \chi_{1,1}(\alpha + r_1^{(k)}\mathcal{H}_{\ell^{(k)}}(\mathbf{id}_k)) - r_1^{(k)}\chi_{1,2},$$
$$(z'_2)^{(k)} = z_2^{(k)} + \chi_{2,1}(\alpha + r_1^{(k)}\mathcal{H}_{\ell^{(k)}}(\mathbf{id}_k)) - r_1^{(k)}\chi_{2,2},$$
$$(w'_1)^{(k)} = w_1^{(k)} + \chi_{1,1}r_2^{(k)}\mathcal{H}_{\ell^{(k)}}(\mathbf{id}_k) - r_2^{(k)}\chi_{1,2},$$
$$(w'_2)^{(k)} = w_2^{(k)} + \chi_{2,1}r_1^{(k)}\mathcal{H}_{\ell^{(k)}}(\mathbf{id}_k) - r_2^{(k)}\chi_{2,2},$$
$$(x'_{1,j})^{(k)} = x_{1,j} + r_1\theta_1\chi_{1,1}, \qquad (y'_{1,j})^{(k)} = y_{1,j} + r_1\theta_1\chi_{2,1} \text{ for } \ell^{(k)} \le j \le h,$$
$$(x'_{2,j})^{(k)} = x_{2,j} + r_2\theta_1\chi_{1,1}, \qquad (y'_{2,j})^{(k)} = y_{2,j} + r_2\theta_1\chi_{2,1} \text{ for } \ell^{(k)} \le j \le h.$$

All the above elements are properly distributed due to the choice of $\mathbf{X}$ and the distribution of the information provided in $\text{Game}_{q,1}$. Note that the public parameters remain unchanged.

Over the basis $(\mathbf{F}, \mathbf{F}^*)$, the coefficient of $\mathbf{f}_1$ in $\widehat{\mathcal{C}}$ is given by $s' = s - t_1\chi_1 - t_2\chi_2$. This quantity s' is the real randomiser for $\widehat{\mathcal{C}}$ since $\mathbf{f}_1$ is the only component vector independent of the message and the identity that it randomises. When s' is the real randomiser the masking factor in C_0 should be $e(P_1, P_2)^{s'\alpha\mathbf{f}_1\cdot\mathbf{f}_1^*}$ causing the messsage M_β to be blinded by a factor of $e(P_1, P_2)^{(t_1\chi_{1,1}+t_2\chi_{2,1})\mathbf{b}_1\cdot\mathbf{b}_1^*}(\in_{\mathrm{U}} \mathbb{G}_T)$. Also the coefficient of $\mathbf{b}_2$ is changed to random coefficient s'' of $\mathbf{f}_2$ and hence $\widehat{\mathcal{C}}$ is indistinguishable from a semi-functional encryption of a random message to a random identity vector. Therefore $\mathcal{E}_{q,1}$ when expressed over $\mathbf{F}, \mathbf{F}^*$ has the same distribution as $\mathcal{E}_{final}$ which means that $\text{Game}_{q,1}$ and Game_{final} are statistically indistinguishable and hence $\Pr[X_{q,1}] = \Pr[X_{final}]$. □

Theorem C.6. *If the* $(\varepsilon_{\text{DDH1}}, t_{\text{DDH1}})$-DDH1 *and* $(\varepsilon_{\text{DDH2}}, t_{\text{DDH2}})$-DDH2 *assumptions hold in* $\mathcal{G}$*, then* DPVS-AHIBE *is* (ε, t, q)-ANO-IND-ID-CPA *secure where*

$$\varepsilon \le \frac{1}{\gamma_p^2}\varepsilon_{\text{DDH1}} + \frac{2q}{\gamma_p^2}\varepsilon_{\text{DDH2}} + \frac{q}{p}$$

and $t_{\text{DDH1}} = t + O(h\rho)$, $t_{\text{DDH2}} = t + O(qh\rho)$ *where* ρ *is the maximum time for one scalar multiplication in either of* $\mathbb{G}_1$,$\mathbb{G}_2$.

Proof. Using lemmas C.1, C.2, C.3, C.4 and C.5, we have for any attacker $\mathscr{A}$ making at most q queries an adversary $\mathscr{B}$ such that

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{ano-ind-cpa}}_{\mathcal{DPVS}\text{-}\mathcal{AHIBE}}(\mathscr{A}) &= \left|\Pr[X_{restricted}] - \frac{1}{2}\right| \\
&= |\Pr[X_{actual}] - \Pr[X_{final}]| \\
&\le |\Pr[X_{actual}] - \Pr[X_{restricted}]| + |\Pr[X_{restricted}] - \Pr[X_0]| \\
&\quad + \sum_{k=1}^{q} (|\Pr[X_{k-1,1}] - \Pr[X_{k,0}]| + |\Pr[X_{k,0}] - \Pr[X_{k,1}]|) \\
&\quad + |\Pr[X_{q,1}] - \Pr[X_{final}]| \\
&\le \frac{1}{\gamma_p^2}\mathsf{Adv}^{DS1}_{\mathcal{V}_4}(\mathscr{B}) + 2q\left(\frac{1}{\gamma_p^2}\mathsf{Adv}^{DS2}_{\mathcal{V}_4}(\mathscr{B})\right) + \frac{q}{p}
\end{aligned}$$

which along with Lemma A.1 implies the theorem. □

D Extending Lewko-Waters IBE in Type-3 Setting

In this section we start with the Lewko-Waters IBE in Type-3 setting and contruct the schemes $\mathcal{LW}$-$\mathcal{HIBE}$ and $\mathcal{LW}$-$\mathcal{AHIBE}$. First we provide the LW-IBE construction followed by the two HIBE schemes along with their security proofs.

D.1 Lewko-Waters IBE

Here we provide the construction of the IBE in the Type-3 setting given by Lewko and Waters [LW10]. We follow the convention used in the previous section for indicating semi-functional components.

Let P_1 and P_2 be random generators of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively. Let $Q_1, U_1 \in_{\mathrm{U}} \mathbb{G}_1$ and $Q_2, U_2 \in \mathbb{G}_2$ be such that $Q_2 \sim Q_1$ and $U_2 \sim U_1$. Choose F_2 at random from $\mathbb{G}_2$, a, v, v' from $\mathbb{Z}_p$ and define $V_2 = vF_2, V_2' = v'F_2$. Define $\tau = v + av'$ so that $\tau F_2 = V_2 + aV_2'$. Identities are elements of $\mathbb{Z}_p$. The public parameters and master secret are given by

$$\begin{aligned}\mathcal{PP} &: (P_1, aP_1, \tau P_1, Q_1, aQ_1, \tau Q_1, U_1, aU_1, \tau U_1, e(P_1,P_2)^\alpha)\\ \mathcal{MSK}&: (\alpha P_2, P_2, V_2, V_2', Q_2, U_2, F_2).\end{aligned}$$

The randomisers for the ciphertext and key are s and w, r_1, r_2 respectively. These are elements of $\mathbb{Z}_p$. For the semi-functional components, μ, π_C and γ, π_K are chosen at random from $\mathbb{Z}_p$. Also elements $V_1', F_1 \in \mathbb{G}_1$ are such that $V_1' \sim V_2'$ and $F_1 \sim F_2$.

Ciphertext:
$C_0 = M \times e(P_1,P_2)^{\alpha s}$
$C_{1,1} = s(\mathsf{id}Q_1 + U_1),\ C_{1,2} = as(\mathsf{id}Q_1 + U_1)\{+\mu\pi_C F_1\},\ C_{1,3} = -\tau s(\mathsf{id}Q_1 + U_1)\{-\mu\pi_C V_1'\}$
$C_{2,1} = sP_1,\ C_{2,2} = asP_1\{+\mu F_1\},\ C_{2,3} = -\tau sP_1\{-\mu V_1'\}$

Key:
$K_{1,1} = wP_2 + r_1V_2\{-a\gamma F_2\},\ K_{1,2} = r_1V_2'\{+\gamma F_2\},\ K_{1,3} = r_1F_2$
$K_{2,1} = \alpha P_2 + w(\mathsf{id}Q_2 + U_2) + r_1V_2\{-a\gamma\pi_K F_2\},\ K_{2,2} = r_2V_2'\{+\gamma\pi_K F_2\},\ K_{2,3} = r_2F_2$

D.2 Construction of $\mathcal{LW}$-$\mathcal{HIBE}$

In this section, we provide a HIBE scheme resulting from a BBG-extension of the LW-IBE scheme provided in Section D.1. A straightforward BBG-type extension would cause problems in simulating key extraction queries specifically while handling the components used for re-randomisation. We introduce some new elements in the key to overcome this problem.

$\mathsf{Setup}\ (\lambda)$: Let h denote the maximum depth of the HIBE. Choose random generators $P_1 \in \mathbb{G}_1$ and $P_2 \in \mathbb{G}_2$; random elements $Q_{1,1}, \ldots, Q_{1,h}, U_1 \in \mathbb{G}_1$ and $Q_{2,1}, \ldots, Q_{2,h}, U_2 \in \mathbb{G}_2$ such that $Q_{2,j} \sim Q_{1,j}$ for all $1 \le j \le h$ and $U_2 \sim U_1$. Let $F_2 \in \mathbb{G}_2$ be chosen at random and v, v' be chosen randomly from $\mathbb{Z}_p$. Set $V_2 = vF_2$, $V_2' = v'F_2$. Pick α, a at random from $\mathbb{Z}_p$. Set $\tau = v + av'$ so that $\tau F_2 = V_2 + aV_2'$.

$$\begin{aligned}\mathcal{PP} &: (P_1, Q_{1,1}, \ldots, Q_{1,h}, U_1, aP_1, aQ_{1,1}, \ldots, aQ_{1,h}, aU_1, \tau P_1, \tau Q_{1,1}, \ldots, \tau Q_{1,h}, \tau U_1, e(P_1,P_2)^\alpha).\\ \mathcal{MSK}&: (\alpha P_2, P_2, V_2, V_2', F_2, Q_{2,1}, \ldots, Q_{2,h}, U_2).\end{aligned}$$

The only quantity of the $\mathcal{MSK}$ which is required to be kept secret by the PKG is αP_2. The rest are made public and are downloaded during delegation.

$\mathsf{Encrypt}$ $(M, \mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{PP})$: Choose random s from $\mathbb{Z}_p$. Let $\mathcal{H}_i(\mathbf{id}) = \mathsf{id}_1 Q_{i,1} + \cdots + \mathsf{id}_\ell Q_{i,\ell} + U_i$ for $i = 1, 2$. The ciphertext is given by $\mathcal{C} = (C_0, C_{1,1}, C_{1,2}, C_{1,3}, C_{2,1}, C_{2,2}, C_{2,3})$ where the elements are computed as follows.

$$C_0 = M \times e(P_1, P_2)^{\alpha s},$$
$$C_{1,1} = s\mathcal{H}_1(\mathbf{id}),\ C_{1,2} = as\mathcal{H}_1(\mathbf{id}),\ C_{1,3} = -\tau s\mathcal{H}_1(\mathbf{id})$$
$$C_{2,1} = sP_1,\ C_{2,2} = asP_1,\ C_{2,3} = -\tau sP_1$$

KeyGen $(\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{MSK}, \mathcal{PP})$: Choose random $w, r_1, r_2, z_{\ell+1}, \ldots, z_h$ from $\mathbb{Z}_p$. The secret key for $\mathbf{id}$, $\mathcal{SK}_{\mathbf{id}}$ is given by $(K_{i,1}, K_{i,2}, K_{i,3}, D_{j,1}, D_{j,2}, D_{j,3})$ for $i = 1, 2$ and $j = \ell + 1, \ldots, h$. So, the key consists of $3(h - \ell + 2)$ group elements. The elements are computed as follows.

$$K_{1,1} = wP_2 + r_1 V_2,\ K_{1,2} = r_1 V_2',\ K_{1,3} = r_1 F_2$$
$$K_{2,1} = \alpha P_2 + w\mathcal{H}_2(\mathbf{id}) + r_2 V_2,\ K_{2,2} = r_2 V_2',\ K_{2,3} = r_2 F_2$$
$$D_{j,1} = wQ_{2,j} + z_j V_2,\ D_{j,2} = z_j V_2',\ D_{j,3} = z_j F_2 \text{ for } \ell + 1 \leq j \leq h.$$

The portion of the key which consists of the K's is used for actual decryption. This part is the same as the key for the IBE scheme with the only change coming in the definition of the identity hash $\mathcal{H}_2(\mathbf{id})$. A straightforward extension of the IBE decryption key would have, for each j, only one D-component D_j to be equal to $wQ_{2,j}$. This would be required for creating the identity-hash for the next level delegation. However, such a direct extension does not work. In the security analysis, relating a decryption key to the problem instance in the second reduction does not work. The solution is to have a V_2 part in the D-component. But, then this requires separate randomisers $z_{\ell+1}, \ldots, z_h$ and also two extra components for cancellation.

$\mathsf{Delegate}$ $(\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{SK}_{\mathbf{id}}, \mathsf{id}_{\ell+1}, \mathcal{PP}, (P_2, V_2, V_2', F_2, Q_{2,1}, \ldots, Q_{2,h}, U_2))$:

Note that the tuple $(P_2, V_2, V_2', F_2, Q_{2,1}, \ldots, Q_{2,h}, U_2)$ is required for key delegation and is made available publicly by the PKG. Define the notation $\mathbf{id} : \mathsf{id}_{\ell+1} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell, \mathsf{id}_{\ell+1})$, i.e., $\mathbf{id} : \mathsf{id}_{\ell+1}$ denotes the tuple obtained by concatenating $\mathsf{id}_{\ell+1}$ to the end of $\mathbf{id}$.

Choose random $r_1', r_2', z_{\ell+2}', \ldots, z_h', w'$ from $\mathbb{Z}_p$. The components of the key for the identity $\mathbf{id} : \mathsf{id}_{\ell+1}$ are computed as follows.

$$K_{1,1} \leftarrow K_{1,1} + r_1' V_2 + w' P_2,\ K_{1,2} \leftarrow K_{1,2} + r_1' V_2',\ K_{1,3} \leftarrow K_{1,3} + r_1' F_2,$$
$$K_{2,1} \leftarrow K_{2,1} + \mathsf{id}_{\ell+1} D_{\ell+1,1} + w'\mathcal{H}_2(\mathbf{id} : \mathsf{id}_{\ell+1}) + r_2' V_2,$$
$$K_{2,2} \leftarrow K_{2,2} + r_2' V_2' + \mathsf{id}_{\ell+1} D_{\ell+1,2},$$
$$K_{2,3} \leftarrow K_{2,3} + r_2' F_2 + \mathsf{id}_{\ell+1} D_{\ell+1,3},$$
$$D_{j,1} \leftarrow D_{j,1} + w' Q_{2,j} + z_j' V_2,\ D_{j,2} \leftarrow D_{j,2} + z_j' V_2',\ D_{j,3} \leftarrow D_{j,3} + z_j' F_2 \text{ for } j = \ell + 2, \ldots, h.$$

The above procedure essentially re-randomises all components of the key. As a result the distribution of a key obtained using delegation is the same as the distribution of a key obtained using the key generation procedure. To note the re-randomisation consider the following change of scalars for the modified key.

$$w \leftarrow w + w';$$
$$r_1 \leftarrow r_1 + r_1';$$
$$r_2 \leftarrow r_2 + \mathsf{id}_{\ell+1} z_{\ell+1} + r_2';$$
$$z_j \leftarrow z_j + z_j' \text{ for } j = \ell + 2, \ldots, h.$$

$\mathsf{Decrypt}\ (\mathcal{C}, \mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{SK}_{\mathbf{id}}, \mathcal{PP})$: Decryption is done as follows.

$$M = C_0 \times \frac{e(C_{1,1}, K_{1,1})e(C_{1,2}, K_{1,2})e(C_{1,3}, K_{1,3})}{e(C_{2,1}, K_{2,1})e(C_{2,2}, K_{2,2})e(C_{2,3}, K_{2,3})}$$

The consistency of the HIBE scheme follows directly from that of LW-IBE since the decryption does not use the additional components used for delegation.

D.3 Proof of Security for *LW-HIBE*

As is typical in the dual-system technique, we first describe the semi-functional ciphertexts and keys.

Semi-functional ciphertext: Let $C'_0, C'_{1,1}, C'_{1,2}, C'_{1,3}, C'_{2,1}, C'_{2,2}, C'_{2,3}$ be ciphertext elements normally generated by the $\mathsf{Encrypt}$ algorithm for message M and identity $\mathbf{id}$. Let V'_1, F_1 be an element of $\mathbb{G}_1$ such that $V'_1 \sim V'_2$ and $F_1 \sim F_2$. Choose $\mu, \pi_C \in \mathbb{Z}_p$ at random. The semi-functional ciphertext generation algorithm will modify the normal ciphertext as: $C_0 = C'_0$, $C_{1,1} = C'_{1,1}$, $C_{2,1} = C'_{2,1}$ and

$$C_{1,2} = C'_{1,2} + \mu\pi_C F_1,\ C_{1,3} = C'_{1,3} - \mu\pi_C V'_1, C_{2,2} = C'_{2,2} + \mu F_1,\ C_{2,3} = C'_{2,3} - \mu V'_1.$$

Semi-functional key: Let $(K'_{i,1}, K'_{i,2}, K'_{i,3}, D_{j,1}, D_{j,2}, D_{j,3})$ for $i = 1, 2$ and $j = \ell+1, \ldots, h$ be secret key components generated by the KeyGen algorithm for identity $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$. Let $\gamma, \pi_K, \gamma_{\ell+1}, \ldots, \gamma_h$ be random elements chosen from $\mathbb{Z}_p$. The semi-functional key generation algorithm will modify the normal key as:

$$\begin{aligned}
&K_{1,1} = K'_{1,1} - a\gamma F_2,\ K_{1,2} = K'_{1,2} + \gamma F_2,\\
&K_{2,1} = K'_{2,1} - a\gamma\pi_K F_2,\ K_{2,2} = K'_{2,2} + \gamma\pi_K F_2,\\
&D_{j,1} = D'_{j,1} - a\gamma_j F_2,\ D_{j,2} = D'_{j,2} + \gamma_j F_2 \ \text{ for } \ell+1 \le j \le h.
\end{aligned}$$

The rest of the components, i.e., $K_{1,3}, K_{2,3}, D_{j,3}$ for $j = \ell+1, \ldots, h$ remain unchanged.

Similar to the security proof of *DPVS-HIBE*, the hybrid argument is over a sequence of $q+3$ games - Game_{actual}, Game_0, ..., Game_q and Game_{final}. The events X_{actual}, X_k and X_{final} are also defined in a similar manner.

For the proof it will be convenient to use the following short-hand: Denote by $h(\mathbf{id})$ the sum $\sum_{j=1}^{\ell} y_{q_j}\mathsf{id}_j + y_u$ and by $g(\mathbf{id})$ the sum $\sum_{j=1}^{\ell} \lambda_j \mathsf{id}_j + \nu$, where $y_{q_1}, \ldots, y_{q_h}, y_u, \lambda_1, \ldots, \lambda_h, \nu$ are random elements of $\mathbb{Z}_p$ and will be defined in the proofs.

The simulation technique extends that of Lewko and Waters. Public parameters and decryption keys have extra components in comparison to LW-IBE. We show that these components can be constructed. For the components which are already present in LW-IBE, the simulation method remains more or less unchanged.

Theorem D.1. *If the* $(\varepsilon_{\mathrm{LW1}}, t')$-LW1, $(\varepsilon_{\mathrm{LW2}}, t')$-LW2 *and* $(\varepsilon_{\mathrm{DBDH}}, t')$-DBDH *assumptions hold, then LW-HIBE is* (ε, t, q)-$\mathsf{IND\text{-}ID\text{-}CPA}$ *secure where*

$$\varepsilon \le \varepsilon_{\mathrm{LW1}} + q\varepsilon_{\mathrm{LW2}} + \varepsilon_{\mathrm{DBDH}}$$

and $t = t' - O(q\rho)$. *Here* ρ *is an upper bound on the time required for one scalar multiplication in* $\mathbb{G}_1$ *or* $\mathbb{G}_2$.

Proof. For any t-time adversary $\mathscr{A}$ against $\mathcal{LW}$-$\mathcal{HIBE}$ in the $\mathsf{ind}-\mathsf{cpa}$, its advantage in winning the game is given by

$$\mathsf{Adv}^{\mathsf{ind}-\mathsf{cpa}}_{\mathcal{LW}\text{-}\mathcal{HIBE}}(\mathscr{A}) = \left|\Pr[X_{actual}] - \frac{1}{2}\right|.$$

We know that $\Pr[X_{final}] = \frac{1}{2}$ and hence we have

$$\begin{aligned}\mathsf{Adv}^{\mathsf{ind}-\mathsf{cpa}}_{\mathcal{LW}\text{-}\mathcal{HIBE}}(\mathscr{A}) &= |\Pr[X_{actual}] - \Pr[X_{final}]| \\ &\leq |\Pr[X_{actual}] - \Pr[X_0]| + \sum_{k=1}^{q} (|\Pr[X_{k-1}] - \Pr[X_k]|) + |\Pr[X_q] - \Pr[X_{final}]| \\ &\leq \varepsilon_{\mathrm{LW1}} + q\varepsilon_{\mathrm{LW2}} + \varepsilon_{\mathrm{DBDH}}\end{aligned}$$

The last inequality follows from the lemmas D.2, D.3 and D.4 that we prove below. In all the lemmas, $\mathscr{A}$ is a t-time adversary against $\mathcal{LW}$-$\mathcal{HIBE}$ and $\mathscr{B}$ is an algorithm running in time t' that interacts with $\mathscr{A}$ and solves one of the three problems LW1, LW2 or DBDH. □

Lemma D.2. $|\Pr[X_{actual}] - \Pr[X_0]| \leq \varepsilon_{\mathrm{LW1}}$.

Proof. The algorithm $\mathscr{B}$ receives the following instance of LW1

$$(F_1, bsF_1, sF_1, aF_1, ab^2F_1, bF_1, b^2F_1, asF_1, b^2sF_1, b^3F_1, b^3sF_1, F_2, bF_2, Z_1).$$

$\mathscr{B}$ has to determine whether $Z_1 = ab^2sF_1$ or $Z_1 \in_{\mathrm{U}} \mathbb{G}_1$. We will call Z_1 "real" in the former case and "random" otherwise. $\mathscr{B}$ simulates the security game as described below.

Set-Up: $\mathscr{B}$ chooses random elements $\alpha, y_g, y'_v, y_{q_1}, \ldots, y_{q_h}, y_u, \lambda_1, \ldots, \lambda_h, \nu$ from $\mathbb{Z}_p$ and sets the following elements.

$$P_1 = b^2F_1 + y_gF_1, Q_{1,j} = \lambda_j(b^2F_1) + y_{q_j}F_1 \text{ for } 1 \leq j \leq h, U_1 = \nu(b^2F_1) + y_uF_1$$
$$V_2 = bF_2, V'_2 = y'_vF_2.$$

This implicitly sets $\tau = b + ay'_v$. The element τP_1 can be computed as $b^3F_1 + y'_v(ab^2F_1) + y_g(bF_1) + y_gy'_v(aF_1)$. The parameter $e(P_1, P_2)^\alpha$ is computed as $(e(b^3F_1 + y_g(bF_1), bF_2)e(P_1, y_gF_2))^\alpha$. The simulator computes the remaining components of the public parameter using α and gives the following public parameters to $\mathscr{A}$.

$$\mathcal{PP} = (P_1, Q_{1,1}, \ldots, Q_{1,h}, U_1, aP_1, aQ_{1,1}, \ldots, aQ_{1,h}, aU_1, \tau P_1, \tau Q_{1,1}, \ldots, \tau Q_{1,h}, \tau U_1, e(P_1, P_2)^\alpha).$$

Phases 1 and 2: $\mathscr{A}$ makes a number of key extract queries. $\mathscr{B}$ does not know P_2 and hence does not have the master secret. The secret key for a query $\mathbf{id}$ is constructed as follows. $\mathscr{B}$ chooses $r'_1, r'_2, z'_{\ell+1}, \ldots, z'_n, w \in \mathbb{Z}_p$ at random and sets $r_1 = r'_1 - wb$, $r_2 = r'_2 - (wg(\mathbf{id}) + \alpha)b$ and $z_j = z'_j - w\lambda_j b$ for $j = \ell + 1, \ldots, h$.

$$K_{1,1} = wy_gF_2 + r'_1(bF_2),\ K_{1,2} = y'_vr_1F_2,\ K_{1,3} = r_1F_2$$

$$K_{2,1} = \alpha y_gF_2 + r'_2(bF_2) + wh(\mathbf{id})F_2,\ K_{2,2} = y'_vr_2F_2,\ K_{2,3} = r_2F_2$$
$$D_{j,1} = wy_{q_j}F_2 + z'_j(bF_2),\ D_{j,2} = y'_vz_jF_2,\ D_{j,3} = z_jF_2 \text{ for } \ell + 1 \leq j \leq h$$

Challenge: $\mathscr{B}$ receives the target identity $\widehat{\mathbf{id}} = (\widehat{\mathsf{id}}_1, \ldots, \widehat{\mathsf{id}}_{\widehat{\ell}})$ and two messages M_0 and M_1 from $\mathscr{A}$. It chooses $\beta \in \{0, 1\}$ at random. $\mathscr{B}$ computes the ciphertext for M_β as follows.

$$C_0 = M_\beta \cdot \left(e(b^3sF_1 + y_g(bsF_1), bF_2)e(b^2sF_1 + y_g(sF_1), y_gF_2)\right)^\alpha = M_\beta \cdot e(P_1, P_2)^{\alpha s}$$

$$C_{1,1} = g(\widehat{\mathbf{id}})(b^2 sF_1) + h(\widehat{\mathbf{id}})(sF_1),\ C_{1,2} = g(\widehat{\mathbf{id}})Z_1 + h(\widehat{\mathbf{id}})(asF_1)$$
$$C_{1,3} = -g(\widehat{\mathbf{id}})(b^3 sF_1) - h(\widehat{\mathbf{id}})(bsF_1) - y'_v g(\widehat{\mathbf{id}})Z_1 - y'_v h(\widehat{\mathbf{id}})(asF_1)$$
$$C_{2,1} = b^2 sF_1 + y_g(sF_1),\ C_{2,2} = Z_1 + y_g(asF_1)$$
$$C_{2,3} = -b^3 sF_1 - y_g(bsF_1) - y'_v Z_1 - y'_v(asF_1).$$

$\mathscr{B}$ returns $\widehat{\mathcal{C}} = (C_0, C_{1,1}, C_{1,2}, C_{1,3}, C_{2,1}, C_{2,2}, C_{2,3})$ to $\mathscr{A}$.

Now if $Z_1 = ab^2 sF_1$ then the challenge ciphertext is normal; otherwise if Z_1 is a random element of $\mathbb{G}_1$ i.e., $Z_1 = (ab^2 s + \mu')F_1$ then the ciphertext is semi-functional. Here μ is implicitly set to μ' and π_C is set to $g(\widehat{\mathbf{id}})$. It is straightforward to verify the well-formedness of $C_{1,2}$,$C_{1,3}$,$C_{2,2}$ and $C_{2,3}$. Note that, to check whether $\widehat{\mathcal{C}}$ is semi-functional or not, $\mathscr{B}$ itself could try to decrypt it with a semi-functional key for $\widehat{\mathbf{id}}$. However since aP_2 is not known to $\mathscr{B}$, it cannot create such a key.

Guess: The adversary returns its guess β' to $\mathscr{B}$.

If Z_1 is real, $\widehat{\mathcal{C}}$ is normal and hence $\mathscr{B}$ simulates Game_{real}. Otherwise, Z_1 is random and $\widehat{\mathcal{C}}$ is semi-functional in which case, $\mathscr{B}$ simulates Game_0. Suppose that $\mathscr{B}$ returns 1 if $\beta = \beta'$ and 0 otherwise. Then it can solve the LW1 problem with advantage

$$\mathsf{Adv}^{\text{LW1}}_{\mathcal{G}}(\mathscr{B}) = |\Pr[\beta = \beta' | Z_1 \text{ is real}] - \Pr[\beta = \beta' | Z_1 \text{ is random}]| = |\Pr[X_{actual}] - \Pr[X_0]|.$$

By definition $\mathsf{Adv}^{\text{LW1}}_{\mathcal{G}}(\mathscr{B}) \leq \varepsilon_{\text{LW1}}$ for any t-time adversary $\mathscr{B}$ and hence follows the lemma. □

Lemma D.3. $|\Pr[X_{k-1}] - \Pr[X_k]| \leq \varepsilon_{\text{LW2}}$ *for* $1 \leq k \leq q$.

Proof. Let $(F_1, dF_1, d^2F_1, bxF_1, dbxF_1, d^2xF_1, F_2, dF_2, bF_2, cF_2, Z_2)$ be the instance of LW2 that $\mathscr{B}$ receives. $\mathscr{B}$'s task is to decide whether Z_2 is real ($Z_2 = bcF_2$) or Z_2 is random ($Z_2 \in_{\text{U}} \mathbb{G}_2$).

Set-Up: $\mathscr{B}$ chooses random elements $\alpha, a, y_v, y_{q_1}, \ldots, y_{q_h}, y_u, \lambda_1, \ldots, \lambda_h, \nu \in_{\text{U}} \mathbb{Z}_p$ and sets the following elements. $P_1 = dF_1, P_2 = dF_2$, $Q_{1,j} = \lambda_j(dF_1) + y_{q_j}F_1$ for $1 \leq j \leq h$, $U_1 = \nu(dF_1) + y_u F_1$, $V_2 = -a(bF_2) + dF_2 + y_v F_2$ and $V'_2 = bF_2$ setting $v = -ab + d + y_v$, $v' = b$ and $\tau = d + y_v$. The element τP_1 can be computed as $\tau P_1 = d^2F_1 + y_v(dF_1)$. The parameters $\tau Q_{1,j}$ for $1 \leq j \leq h$ and τU_1 can be computed as $\tau Q_{1,j} = \lambda_j(d^2F_1) + y_{q_j}(dF_1) + y_v\lambda_j(dF_1) + y_v y_{q_j}F_1$ and $\tau U_1 = \nu(d^2F_1) + y_u(dF_1) + y_v\nu(dF_1) + y_v y_u F_1$. The remaining parameters required to provide $\mathcal{PP}$ to $\mathscr{A}$ are computed using a, α and other elements of the problem instance.

Phases 1 and 2: The key extraction queries for identities $\mathbf{id}_1, \ldots, \mathbf{id}_q$ are answered in the following way. For $i < k$, a semi-functional key is returned and for $i > k$ a normal key is returned. This can be done since $\mathscr{B}$ has the master secret and also knows a.

For $i = k$, $\mathscr{B}$ computes the components of the key using the problem instance in the following manner. Let $\mathbf{id}_k = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$. $\mathscr{B}$ chooses $w', r'_2, z'_{\ell+1}, \ldots, z'_h$ randomly from $\mathbb{Z}_p$.

$$K_{1,1} = w'P_2 - aZ_2 + y_v(cF_2),\ K_{1,2} = Z_2,\ K_{1,3} = cF_2$$
$$K_{2,1} = \alpha P_2 + w'(g(\mathbf{id}_k)(dF_2) + h(\mathbf{id}_k)F_2) + r'_2 V_2 - ag(\mathbf{id}_k)Z_2 + y_v g(\mathbf{id}_k)(cF_2) - h(\mathbf{id}_k)cF_2$$
$$K_{2,2} = r'_2 V'_2 + g(\mathbf{id}_k)Z_2,\ K_{2,3} = r'_2 F_2 + g(\mathbf{id}_k)(cF_2)$$

and for $j = \ell + 1, \ldots, h$, set

$$D_{j,1} = w'Q_{2,j} + z'_j V_2 - y_{q_j}(cF_2) - a\lambda_j Z_2 + y_v\lambda_j(cF_2)$$

$$D_{j,2} = z'_j V'_2 + \lambda_j Z_2,\ D_{j,3} = z'_j F_2 + \lambda_j (cF_2)$$

thus implicitly setting $r_1 = c$, $r_2 = r'_2 + g(\mathbf{id}_k)c$, $w = w' - c$, $\pi_K = g(\mathbf{id}_k)$ and $z_j = z'_j + \lambda_j c$ for $\ell+1 \leq j \leq h$.

$\mathscr{B}$ returns $\mathcal{SK}_{\mathbf{id}_k} = (K_{i,1}, K_{i,2}, K_{i,3}, D_{j,1}, D_{j,2}, D_{j,3})$, $i = 1, 2$ and $j = 1, 2, \ell+1, \ldots, h$ as the key for $\mathbf{id}_k$. If $Z_2 = bcF_2$ then the key for $\mathbf{id}_k$ will be normal and if $Z_2 = (bc + \gamma')F_2$ the key will be semi-functional with $\gamma = \gamma'$, $\gamma_j = \lambda_j \gamma'$ for $\ell + 1 \leq j \leq h$. $\mathscr{B}$ could also check whether $\mathcal{SK}_{\mathbf{id}_k}$ is normal or semi-functional by creating a semi-functional ciphertext for id_k. Since $V'_1 = bF_1$ is not available to $\mathscr{B}$, the only way of doing this will lead to π_C being the same as π_K (such a ciphertext can be constructed in a manner similar to creating the challenge ciphertext). The ciphertext-key pair will be nominally semi-functional and thus provides no information to $\mathscr{B}$.

Challenge: $\mathscr{B}$ receives two messages M_0, M_1 and a challenge identity $\widehat{\mathbf{id}} = (\widehat{\mathsf{id}}_1, \ldots, \widehat{\mathsf{id}}_{\widehat{\ell}})$ during the challenge phase. It chooses $\beta \in_{\mathrm{U}} \{0, 1\}$, generates the challenge ciphertext as shown below.

$$C_0 = M_\beta \cdot e(dbxF_1, dF_2)^\alpha$$

$$C_{1,1} = g(\widehat{\mathbf{id}})(dbxF_1) + h(\widehat{\mathbf{id}})(bxF_1)$$

$$C_{1,2} = ag(\widehat{\mathbf{id}})(dbxF_1) + ah(\widehat{\mathbf{id}})(bxF_1) - g(\widehat{\mathbf{id}})(d^2 xF_1)$$

$$C_{1,3} = -y_v g(\widehat{\mathbf{id}})(dbxF_1) - h(\widehat{\mathbf{id}})(dbxF_1) - y_v h(\widehat{\mathbf{id}})(bxF_1)$$

$$C_{2,1} = dbxF_1,\ C_{2,2} = a(dbxF_1) - d^2 xF_1,\ C_{2,3} = -y_v(dbxF_1).$$

This sets $s = bx$, $\mu = -d^2 x$ and $\pi_C = g(\widehat{\mathbf{id}})$.

Since $\lambda_1, \ldots, \lambda_h$ and ν are chosen uniformly at random from $\mathbb{Z}_p$, $\pi_K = \lambda_1 \mathsf{id}_1 + \cdots + \lambda_\ell \mathsf{id}_\ell + \nu$ and $\pi_C = \lambda_1 \widehat{\mathsf{id}}_1 + \cdots + \lambda_{\widehat{\ell}} \widehat{\mathsf{id}}_{\widehat{\ell}} + \nu$ are independent and uniformly distributed.

Guess: $\mathscr{A}$ returns a bit β' as its guess for β.

When the instance is real, $\mathscr{B}$ simulates Game_{k-1} and otherwise simulates Game_k. $\mathscr{B}$ returns 1 if $\mathscr{A}$ wins the game i.e., $\beta = \beta'$; otherwise it returns 0. Hence, $\mathscr{B}$ can solve the LW2 instance with advantage

$$\mathsf{Adv}^{\mathrm{LW2}}_{\mathcal{G}}(\mathscr{B}) = |\Pr[\beta = \beta' | Z_2 \text{ is real}] - \Pr[\beta = \beta' | Z_2 \text{ is random}]| = |\Pr[X_{k-1}] - \Pr[X_k]|.$$

from which the statement of the lemma clearly follows. □

Lemma D.4. $|\Pr[X_q] - \Pr[X_{final}]| \leq \varepsilon_{\mathrm{DBDH}}$.

Proof. $\mathscr{B}$ receives $(F_1, aF_1, bF_1, sF_1, F_2, aF_2, bF_2, sF_2, Z_T)$ as an instance of the DBDH problem where $Z_T = e(F_1, F_2)^{abs}$ (real) or $Z_T \in_{\mathrm{U}} \mathbb{G}_T$ (random).

Set-Up: With $y_g, y_v, y'_v, y_{q_1}, \ldots, y_{q_h}, y_u$ chosen at random from $\mathbb{Z}_p$, $\mathscr{B}$ sets the parameters as

$$P_1 = y_g F_1, P_2 = y_g F_2, aP_1 = y_g(aF_1), V_2 = y_v F_2, V'_2 = y'_v F_2,\ \tau P_1 = y_g y_v F_1 + y_g y'_v (aF_1)$$

$$Q_{1,j} = y_{q_j} F_1 \text{ for } 1 \leq j \leq h, U_1 = y_u F_1, e(P_1, P_2)^\alpha = e(aF_1, bF_2)^{{y_g}^2}$$

implicitly setting $\alpha = ab$ and $\tau = y_v + ay'_v$. The remaining parameters can be computed easily. $\mathscr{B}$ returns $\mathcal{PP}$ to $\mathscr{A}$.

Phases 1 and 2: When $\mathscr{A}$ asks for the secret key for the i'th identity $\mathbf{id}_i = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$, $\mathscr{B}$ chooses at random $w, r_1, r_2, z_{\ell+1}, \ldots z_h \in \mathbb{Z}_p$ and $\gamma', \gamma, \gamma_{\ell+1}, \ldots, \gamma_h \in \mathbb{Z}_p$ and computes a semi-functional key for $\mathbf{id}_i$ as follows.

$$K_{1,1} = wP_2 + r_1 V_2 - \gamma(aF_2),\ K_{1,2} = r_1 V'_2 + \gamma F_2,\ K_{1,3} = r_1 F_2$$

$$K_{2,1} = \gamma'(aF_2) + wh(\mathbf{id}_i)F_2 + r_2V_2,\ K_{2,2} = r_2V_2' + y_g(bF_2) - \gamma'F_2,\ K_{2,3} = r_2F_2,$$
$$D_{j,1} = wQ_{2,j} + z_jV_2 - \gamma_j(aF_2),\ D_{j,2} = z_jV_2' + \gamma_jF_2,\ D_{j,3} = z_jF_2 \text{ for } \ell+1 \le j \le h.$$

Here $\gamma' = by_g - \gamma\pi_K$ and is implicit in the construction. Observe that $\mathscr{B}$ does not know α and hence cannot create a normal key.

Challenge: $\mathscr{B}$ receives the challenge identity $\widehat{\mathbf{id}} = (\widehat{\mathsf{id}}_1, \ldots, \widehat{\mathsf{id}}_{\widehat{\ell}})$ and two messages M_0 and M_1 from $\mathscr{A}$. It chooses $\beta \in \{0,1\}$ and $\mu_1, \mu_2 \in \mathbb{Z}_p$ at random and generates a semi-functional challenge ciphertext as follows.

$$C_0 = M_\beta \times Z_T$$
$$C_{1,1} = h(\widehat{\mathbf{id}})sF_1,\ C_{1,2} = h(\widehat{\mathbf{id}})\mu_1F_1,\ C_{1,3} = -y_vh(\widehat{\mathbf{id}})(sF_1) - y_v'h(\widehat{\mathbf{id}})\mu_1F_1$$
$$C_{2,1} = y_g(sF_1),\ C_{2,2} = \mu_2F_1,\ C_{2,3} = -y_gy_v(sF_1) - y_v'\mu_2F_1$$

with $\mu_1 = as + \mu$, $\mu_2 = asy_g + \mu$ and $\pi_C = h(\widehat{\mathbf{id}})$.

The challenge ciphertext $\widehat{\mathcal{C}} = (C_0, C_{1,1}, C_{1,2}, C_{1,3}, C_{2,1}, C_{2,2}, C_{2,3})$ is returned to $\mathscr{A}$.

Guess: $\mathscr{A}$ returns its guess β' of β.

It is clear that $\widehat{\mathcal{C}}$ is a semi-functional encryption of M_β when $Z_T = e(P_1,P_2)^{abs}$. And when $Z_T \in_{\mathrm{U}} \mathbb{G}_T$ $\widehat{\mathcal{C}}$ would be a semi-functional encryption of a random message. Hence $\mathscr{B}$ simulates Game_q or Game_{final} according to Z_T being real or random respectively. If the algorithm $\mathscr{B}$ returns 1 when $\beta = \beta'$ and 0 otherwise, it can solve the DBDH instance with advantage

$$\mathsf{Adv}_{\mathcal{G}}^{\mathrm{DBDH}}(\mathscr{B}) = |\Pr[\beta = \beta' | Z_T \text{ is real}] - \Pr[\beta = \beta' | Z_T \text{ is random}]| = |\Pr[X_q] - \Pr[X_{final}]|.$$

□

D.4 Construction of *LW-AHIBE*

In this section, we extend *LW-HIBE* to achieve anonymity thus obtaining the scheme *LW-AHIBE*. The technique is similar to the one we used to construct *DPVS-AHIBE*. Now we provide the construction and the security proof is provided in Section D.5.

Setup (λ): Let h denote the maximum depth of the HIBE. Choose random generators $P_1 \in \mathbb{G}_1$ and $P_2 \in \mathbb{G}_2$; random elements $Q_{1,1}, \ldots, Q_{1,h}, U_1 \in \mathbb{G}_1$ and $Q_{2,1}, \ldots, Q_{2,h}, U_2 \in \mathbb{G}_2$ such that $Q_{2,j} \sim Q_{1,j}$ for all $1 \le j \le h$ and $U_2 \sim U_1$. Let $F_2 \in \mathbb{G}_2$ be chosen at random and v, v' be chosen randomly from $\mathbb{Z}_p$. Set $V_2 = vF_2$, $V_2' = v'F_2$. Pick α, a at random from $\mathbb{Z}_p$. Set $\tau = v + av'$ so that $\tau F_2 = V_2 + aV_2'$.

$\mathcal{PP}$: $(P_1, Q_{1,1}, \ldots, Q_{1,h}, U_1, aP_1, aQ_{1,1}, \ldots, aQ_{1,h}, aU_1, \tau P_1, \tau Q_{1,1}, \ldots, \tau Q_{1,h}, \tau U_1, e(P_1,P_2)^\alpha)$.
$\mathcal{MSK}$: $(\alpha P_2, P_2, V_2, V_2', F_2, Q_{2,1}, \ldots, Q_{2,h}, U_2)$.

The only quantity of the $\mathcal{MSK}$ which is required to be kept secret by the PKG is αP_2. The remaining components of the $\mathcal{MSK}$ are not required to be kept secret. These are required for key delegation and can be maintained by the PKG on its public database. So, in practice the PKG would choose to make both $\mathcal{PP}$ and the tuple $(P_2, V_2, V_2', F_2, Q_{2,1}, \ldots, Q_{2,h}, U_2)$ public. During encryption, only $\mathcal{PP}$ would be downloaded, whereas for key delegation the tuple $(P_2, V_2, V_2', F_2, Q_{2,1}, \ldots, Q_{2,h}, U_2)$ would be downloaded.

$\mathsf{Encrypt}$ $(M, \mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{PP})$: Choose $s \xleftarrow{\mathrm{U}} \mathbb{Z}_p$. Let $\mathcal{H}_i(\mathbf{id}) = \mathsf{id}_1Q_{i,1} + \cdots + \mathsf{id}_\ell Q_{i,\ell} + U_i$ for $i = 1, 2$. The ciphertext is given by $\mathcal{C} = (C_0, C_{1,1}, C_{1,2}, C_{1,3}, C_{2,1}, C_{2,2}, C_{2,3})$ where the elements are computed as follows.

$C_0 = M \times e(P_1, P_2)^{\alpha s}$,
$C_{1,1} = s\mathcal{H}_1(\mathbf{id})$, $C_{1,2} = as\mathcal{H}_1(\mathbf{id})$, $C_{1,3} = -\tau s\mathcal{H}_1(\mathbf{id})$
$C_{2,1} = sP_1$, $C_{2,2} = asP_1$, $C_{2,3} = -\tau sP_1$

KeyGen $(\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{MSK}, \mathcal{PP})$: Choose $w_1, w_2, r_1, r_2, r_3, r_4, z_{1,\ell+1}, \ldots, z_{1,h}, z_{2,\ell+1}, \ldots, z_{2,h} \xleftarrow{\mathrm{U}} \mathbb{Z}_p$. The secret key for $\mathbf{id}$, $\mathcal{SK}_{\mathbf{id}}$ is given by $((K_{1,i}), (K_{2,i}), (D_{j,i}), (J_{1,i}), (J_{2,i}), (E_{j,i}))$ for $i = 1, 2, 3$ and $j = \ell + 1, \ldots, h$. So, the key consists of $6(n - \ell + 2)$ group elements. The elements are computed as follows.

$K_{1,1} = w_1 P_2 + r_1 V_2$, $K_{1,2} = r_1 V_2'$, $K_{1,3} = r_1 F_2$
$K_{2,1} = \alpha P_2 + w_1 \mathcal{H}_2(\mathbf{id}) + r_2 V_2$, $K_{2,2} = r_2 V_2'$, $K_{2,3} = r_2 F_2$
$D_{j,1} = w_1 Q_{2,j} + z_{1,j} V_2$, $D_{j,2} = z_{1,j} V_2'$, $D_{j,3} = z_{1,j} F_2$ for $\ell + 1 \leq j \leq h$

$J_{1,1} = w_2 P_2 + r_3 V_2$, $J_{1,2} = r_3 V_2'$, $J_{1,3} = r_3 F_2$
$J_{2,1} = w_2 \mathcal{H}_2(\mathbf{id}) + r_4 V_2$, $J_{2,2} = r_4 V_2'$, $J_{2,3} = r_4 F_2$
$E_{j,1} = w_2 Q_{2,j} + z_{2,j} V_2$, $E_{j,2} = z_{2,j} V_2'$, $E_{j,3} = z_{2,j} F_2$ for $\ell + 1 \leq j \leq h$.

$\mathsf{Delegate}$ $(\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{SK}_{\mathbf{id}}, \mathsf{id}_{\ell+1}, \mathcal{PP})$: Choose random w_1', w_2' from $\mathbb{Z}_p^\times$. The components of the key for the identity $\mathbf{id} : \mathsf{id}_{\ell+1}$ are computed as follows.

$K_{1,i} \leftarrow K_{1,i} + w_1' J_{1,i}$ for $i = 1, 2, 3$
$K_{2,i} \leftarrow K_{2,i} + \mathsf{id}_{\ell+1} D_{\ell+1,i} + w_1'(J_{2,i} + \mathsf{id}_{\ell+1} E_{\ell+1,i})$ for $i = 1, 2, 3$
$D_{j,i} \leftarrow D_{j,i} + w_1' E_{j,i}$ for $j = \ell + 2, \ldots, h$ and $i = 1, 2, 3$

$J_{1,i} \leftarrow w_2' J_{1,i}$ for $i = 1, 2, 3$
$J_{2,i} \leftarrow w_2'(J_{2,i} + \mathsf{id}_{\ell+1} E_{\ell+1,i})$ for $i = 1, 2, 3$
$E_{j,i} \leftarrow w_2' E_{j,i}$ for $j = \ell + 2, \ldots, h$ and $i = 1, 2, 3$

The above procedure essentially re-randomises all components of the key. As a result the distribution of a key obtained using delegation is the same as the distribution of a key obtained using the key generation procedure. To note the re-randomisation consider the following change of scalars for the modified key.

$w_1 \leftarrow w_1 + w_1' w_2$;
$r_1 \leftarrow r_1 + w_1' r_3$;
$r_2 \leftarrow r_2 + \mathsf{id}_{\ell+1} z_{1,\ell+1} + w_2'(r_4 + \mathsf{id}_{\ell+1} z_{2,\ell+1})$;
$z_{1,j} \leftarrow z_{1,j} + w_1' z_{2,j+1}$ for $j = \ell + 2, \ldots, h$

$w_2 \leftarrow w_2' w_2$
$r_3 \leftarrow w_2' r_3$;
$r_4 \leftarrow w_2'(r_4 + \mathsf{id}_{\ell+1} z_{2,\ell+1})$;
$z_{2,j} \leftarrow w_2' z_{2,j}$ for $j = \ell + 2, \ldots, h$

These new randomisers are properly distributed since $w_1', w_2' \in \mathbb{Z}_p^\times$ and before re-randomisation, the scalars are uniformly and independently distributed.

Decrypt $(\mathcal{C}, \mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{SK}_{\mathbf{id}}, \mathcal{PP})$: Decryption is done as follows.

$$M = C_0 \times \frac{e(C_{1,1}, K_{1,1})e(C_{1,2}, K_{1,2})e(C_{1,3}, K_{1,3})}{e(C_{2,1}, K_{2,1})e(C_{2,2}, K_{2,2})e(C_{2,3}, K_{2,3})}$$

D.5 Security of *LW-AHIBE*

Semi-functional ciphertexts are identical to those of *LW-HIBE* in Section D.3. Semi-functional keys and partial semi-functional keys are defined here. We will call the elements $(K_{1,i}), (K_{2,i}), (D_{j,i})$ for $i = 1, 2, 3$, $j = \ell + 1, \ldots, h$ of the key as the first set of components and the rest of the elements in the key as the second set of components.

Semi-functional key: Let $((K_{1,i}), (K_{2,i}), (D_{j,i}), (J_{1,i}), (J_{2,i}), (E_{j,i}))$ for $i = 1, 2, 3$ and $j = \ell + 1, \ldots, h$ be secret key components generated by the KeyGen algorithm for identity $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$. Let $\gamma, \pi_K, (\gamma_j), \eta, \sigma_K, (\eta_j)$, $j = \ell + 1, \ldots, h$ be uniform random elements chosen from $\mathbb{Z}_p$. The semi-functional key generation algorithm will modify the normal key as:

$$K_{1,1} = K_{1,1} - a\gamma F_2,\ K_{1,2} = K_{1,2} + \gamma F_2,$$
$$K_{2,1} = K_{2,1} - a\gamma\pi_K F_2,\ K_{2,2} = K_{2,2} + \gamma\pi_K F_2,$$
$$D_{j,1} = D_{j,1} - a\gamma_j F_2,\ D_{j,2} = D_{j,2} + \gamma_j F_2 \ \text{ for } \ell + 1 \le j \le h$$

$$J_{1,1} = J_{1,1} - a\eta F_2,\ J_{1,2} = J_{1,2} + \eta F_2,$$
$$J_{2,1} = J_{2,1} - a\eta\sigma_K F_2,\ J_{2,2} = J_{2,2} + \eta\sigma_K F_2,$$
$$E_{j,1} = E_{j,1} - a\eta_j F_2,\ E_{j,2} = E_{j,2} + \eta_j F_2 \ \text{ for } \ell + 1 \le j \le h.$$

The rest of the components remain unchanged.

Partial semi-functional key: In a partial semi-functional key the second set of components of the key are normal and the first set alone is semi-functional.

The proof strategy is the same as that of *DPVS-AHIBE*. Security will be argued over a sequence of $2q+4$ games - $\mathrm{Game}_{actual}, \mathrm{Game}_0, (\mathrm{Game}_{k,0}, \mathrm{Game}_{k,1})_{1 \le k \le q}, \mathrm{Game}_{M-random}, \mathrm{Game}_{final}$. The first $2q + 2$ games are defined in a manner similar to the proof of *DPVS-AHIBE*. In $\mathrm{Game}_{M-random}$, the challenge ciphertext is a semi-functional encryption of a random message under the challnege identity. Game_{final} is similar to $\mathrm{Game}_{M-random}$ except that the challenge ciphertext is a semi-functional encryption of a random message under a random identity vector. The notation X_{actual}, $X_{k,0}$, $X_{k,1}$, $X_{M-random}$ and X_{final} have the usual meaning.

Theorem D.5. *If the* $(\varepsilon_{\mathrm{LW1}}, t')$-LW1, $(\varepsilon_{\mathrm{LW2}}, t')$-LW2, $(\varepsilon_{\mathrm{DBDH}}, t')$-DBDH *and* $(\varepsilon_{\mathrm{A1}}, t')$-A1 *assumptions hold, then LW-HIBE is* (ε, t, q)-IND-ID-CPA *secure where*

$$\varepsilon \le \varepsilon_{\mathrm{LW1}} + 2q\varepsilon_{\mathrm{LW2}} + \varepsilon_{\mathrm{DBDH}} + \varepsilon_{\mathrm{A1}}$$

and $t = t' - O(q\rho)$*, where* ρ *is an upper bound on the time required for one scalar multiplication in* $\mathbb{G}_1$ *or* $\mathbb{G}_2$*.*

Proof. For any t-time adversary $\mathscr{A}$ against $\mathcal{LW}$-$\mathcal{HIBE}$ in the $\mathsf{ind}-\mathsf{cpa}$, its advantage in winning the game is given by

$$\mathsf{Adv}^{\mathsf{ind}-\mathsf{cpa}}_{\mathcal{LW}\text{-}\mathcal{HIBE}}(\mathscr{A}) = \left|\Pr[X_{actual}] - \frac{1}{2}\right|.$$

We know that $\Pr[X_{final}] = \frac{1}{2}$ and hence we have

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{ind}-\mathsf{cpa}}_{\mathcal{LW}\text{-}\mathcal{HIBE}}(\mathscr{A}) &= |\Pr[X_{actual}] - \Pr[X_{final}]| \\
&\leq |\Pr[X_{actual}] - \Pr[X_0]| + \sum_{k=1}^{q} (|\Pr[X_{k-1,1}] - \Pr[X_{k,0}]|) \\
&\quad + \sum_{k=1}^{q} (|\Pr[X_{k,0}] - \Pr[X_{k,1}]|) + |\Pr[X_q] - \Pr[X_{M-random}]| \\
&\quad + |\Pr[X_{M-random}] - \Pr[X_{final}]| \\
&\leq \varepsilon_{\mathrm{LW1}} + 2q\varepsilon_{\mathrm{LW2}} + \varepsilon_{\mathrm{DBDH}} + \varepsilon_{\mathrm{A1}}
\end{aligned}$$

The last inequality follows from the lemmas D.6, D.7, D.8, D.9 and D.10. In all the lemmas, $\mathscr{A}$ is a t-time adversary against $\mathcal{LW}$-$\mathcal{HIBE}$ and $\mathscr{B}$ is an algorithm running in time t' that interacts with $\mathscr{A}$ and solves one of the three problems LW1, LW2, DBDH or A1. □

Lemma D.6. $|\Pr[X_{actual}] - \Pr[X_0]| \leq \varepsilon_{\mathrm{LW1}}$.

The proof is same as that of Lemma D.2 and so we do not provide it here.

Lemma D.7. $|\Pr[X_{k-1,1}] - \Pr[X_{k,0}]| \leq \varepsilon_{\mathrm{LW2}}$ *for* $1 \leq k \leq q$.

The prood follows in a similar manner as that of Lemma D.3. Instance is embedded in the first set of components of the k-th key. Since $\mathscr{B}$ knows the master secret, the second set of components can be generated easily.

Lemma D.8. $|\Pr[X_{k,0}] - \Pr[X_{k,1}]| \leq \varepsilon_{\mathrm{LW2}}$ *for* $1 \leq k \leq q$.

Proof is similar to Lemma D.3 except that π_K is replaced by σ_K and the first set of components of the k-th key are generated according to the partial semi-functional key generation algorithm.

Lemma D.9. $|\Pr[X_q] - \Pr[X_{M-random}]| \leq \varepsilon_{\mathrm{DBDH}}$.

Proof. The proof is more or less similar Lemma D.4. We will only indicate the differences here. $\mathscr{B}$ receives $(F_1, aF_1, bF_1, sF_1, F_2, aF_2, bF_2, sF_2, Z_T)$ as an instance of the DBDH problem where $Z_T = e(F_1, F_2)^{abs}$ (real) or $Z_T \in_{\mathrm{U}} \mathbb{G}_T$ (random). The **Set-Up**, **Challenge** and **Guess** phases are the same. Only the responses to key request queries are answered differently.

Phases 1 and 2: When $\mathscr{A}$ asks for the secret key for the i'th identity $\mathbf{id}_i = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$, $\mathscr{B}$ chooses at random $w, r_1, r_2, (z_{1,j}, z_{2,j}), \gamma', \gamma, (\gamma_j), \eta, (\eta_j), \sigma_K \in \mathbb{Z}_p$ where $\ell + 1 \leq j \leq h$ and computes a semi-functional key for $\mathbf{id}_i$ as follows.

$$K_{1,1} = w_1 P_2 + r_1 V_2 - \gamma(aF_2),\ K_{1,2} = r_1 V_2' + \gamma F_2,\ K_{1,3} = r_1 F_2$$

$$K_{2,1} = \gamma'(aF_2) + w_1 h(\mathbf{id}_i) F_2 + r_2 V_2,\ K_{2,2} = r_2 V_2' + y_g(bF_2) - \gamma' F_2,\ K_{2,3} = r_2 F_2,$$

$$D_{j,1} = w_1 Q_{2,j} + z_{1,j} V_2 - \gamma_j(aF_2),\ D_{j,2} = z_{1,j} V_2' + \gamma_j F_2,\ D_{j,3} = z_{1,j} F_2 \text{ for } \ell + 1 \leq j \leq h.$$

$$J_{1,1} = w_2 P_2 + r_3 V_2 - \eta(aF_2),\ J_{1,2} = r_3 V_2' + \eta F_2,\ J_{1,3} = r_3 F_2$$

$$J_{2,1} = w_2 h(\mathbf{id}_i) F_2 + r_4 V_2 - \eta\sigma_K a F_2,\ J_{2,2} = r_4 V_2' + \eta\sigma_K F_2,\ J_{2,3} = r_4 F_2,$$

$$E_{j,1} = w_2 Q_{2,j} + z_{2,j} V_2 - \eta_j(aF_2),\ E_{j,2} = z_{2,j} V_2' + \eta_j F_2,\ E_{j,3} = z_{2,j} F_2 \text{ for } \ell + 1 \le j \le h.$$

Here $\gamma' = by_g - \gamma\pi_K$ and is implicit in the construction. Observe that $\mathscr{B}$ does not know α and hence cannot create a normal key. Following the same reasoning about the distribution of the ciphertext as given in Lemma D.4, we have

$$\mathsf{Adv}_{\mathcal{G}}^{\mathrm{DBDH}}(\mathscr{B}) = |\Pr[\beta = \beta' | Z_T \text{ is real}] - \Pr[\beta = \beta' | Z_T \text{ is random}]| = |\Pr[X_q] - \Pr[X_{M-random}]|.$$

□

Lemma D.10. $|\Pr[X_{M-random}] - \Pr[X_{final}]| \le \varepsilon_{\mathrm{A1}}$.

Proof. Let $(\mathcal{G}, F_1, dF_1, dxF_1, sdF_1, F_2, dF_2, xzF_2, dzF_2, Z_1)$ be the instance of A1 provided to $\mathscr{B}$. The game is simulated as follows.

Set-Up: With $\alpha, a, y_g, y_v, y_v', y_{q_1}, \ldots, y_{q_h}, y_u$ chosen at random from $\mathbb{Z}_p$, $\mathscr{B}$ sets the parameters as

$$P_1 = y_g dF_1, V_2 = y_v F_2, V_2' = y_v' F_2$$

$$Q_{1,j} = y_{q_j} dxF_1 \text{ for } 1 \le j \le h, U_1 = y_u dxF_1, e(P_1, P_2)^\alpha = e(dF_1, dF_2)^{y_g^2 \alpha}$$

implicitly setting $P_2 = y_g dF_2$ and $\tau = y_v + a y_v'$. The remaining parameters can be computed easily. $\mathscr{B}$ returns $\mathcal{PP}$ to $\mathscr{A}$.

Key Extraction Phases 1 and 2: $\mathscr{B}$ picks $r_1, r_2, r_3, r_4, w_1, w_2, w_1', w_2', \gamma, (\gamma_j), \eta, (\eta_j) \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ and $c \xleftarrow{\mathrm{U}} \mathbb{Z}_p^\times$. It then computes the key for the i-th identity vector $\mathbf{id}_i = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$ as follows.

$$K_{1,1} = y_g dzF_2 + r_1 V_2 - a\gamma F_2,\ K_{1,2} = r_1 V_2' + \gamma F_2,\ K_{1,3} = r_1 F_2$$

$$K_{2,1} = \alpha P_2 + w_1' h(\mathbf{id}_i) dF_2 + r_2 V_2,\ K_{2,2} = r_2 V_2' + a^{-1}(xzF_2 - w_1' F_2),\ K_{2,3} = r_2 F_2,$$

$$D_{j,1} = w_1' y_{q_j} dF_2 + z_{1,j} V_2,\ D_{j,2} = z_{1,j} V_2' + y_{q_j} a^{-1}(xzF_2 - w_1' F_2),\ D_{j,3} = z_{1,j} F_2 \text{ for } \ell + 1 \le j \le h.$$

$$J_{1,1} = c y_g dzF_2 P_2 + r_3 V_2 - a\eta F_2,\ J_{1,2} = r_3 V_2' + \eta F_2,\ J_{1,3} = r_3 F_2$$

$$J_{2,1} = w_2' h(\mathbf{id}_i) F_2 + r_4 V_2,\ J_{2,2} = r_4 V_2' + a^{-1}(cxzF_2 - w_2' F_2),\ J_{2,3} = r_4 F_2,$$

$$E_{j,1} = w_2' Q_{2,j} + z_{2,j} V_2,\ E_{j,2} = z_{2,j} V_2' + y_{q_j} a^{-1}(cxzF_2 - w_1' F_2),\ E_{j,3} = z_{2,j} F_2 \text{ for } \ell + 1 \le j \le h.$$

setting $w_1 = z$, $w_1' = xw_1 - a\sigma_K\gamma$, $w_2 = cz$, $w_1' = xw_1 - a\sigma_K\eta$, $\gamma_j = y_{q_j}\pi_K\gamma$, and $\eta_j = y_{q_j}\pi_K\eta$. Note that since $\mathscr{B}$ does not know dxF_2, it cannot create a normal key for any identity which rules out the possiblity of $\mathscr{B}$ solving the A1 problem on its own by generating a normal key for the challenge identity.

Challenge: $\mathscr{B}$ receives the challenge identity $\widehat{\mathbf{id}} = (\widehat{\mathsf{id}}_1, \ldots, \widehat{\mathsf{id}}_{\widehat{\ell}})$ and two messages M_0 and M_1 from $\mathscr{A}$. It chooses $\beta \in \{0, 1\}$ and $\mu, \pi_C \in \mathbb{Z}_p$ at random and generates a semi-functional challenge ciphertext as follows.

$$C_0 \xleftarrow{\mathrm{U}} \mathbb{G}_T$$

$$C_{1,1} = h(\widehat{\mathbf{id}}) Z_1,\ C_{1,2} = h(\widehat{\mathbf{id}})(aZ_1) + \mu\pi_C F_1,\ C_{1,3} = -h(\widehat{\mathbf{id}})(\tau Z_1) - \mu\pi_C V_1',$$

$$C_{2,1} = y_g(sdF_1),\ C_{2,2} = ay_q(sdF_1) + \mu F_1,\ C_{2,3} = -\tau(sdF_1) - \mu V_1'.$$

The challenge ciphertext $\widehat{\mathcal{C}} = (C_0, C_{1,1}, C_{1,2}, C_{1,3}, C_{2,1}, C_{2,2}, C_{2,3})$ is returned to $\mathscr{A}$.

Guess: $\mathscr{A}$ returns its guess β' of β.

It is clear that $\widehat{\mathcal{C}}$ is a semi-functional encryption of a random message under $\widehat{\mathbf{id}}$ when Z_1 is real i.e., when $Z_1 = sdxF_1$. When Z_1 is random, $\widehat{\mathcal{C}}$ will be a semi-functional encryption of a random message under a random identity vector since the components $C_{1,1}, C_{1,2}, C_{1,3}$ are random quantities. If the algorithm $\mathscr{B}$ returns 1 when $\beta = \beta'$ and 0 otherwise, it can solve the A1 instance with advantage

$$\mathsf{Adv}_{\mathcal{G}}^{\mathrm{DBDH}}(\mathscr{B}) = |\Pr[\beta = \beta' | Z_1 \text{ is real}] - \Pr[\beta = \beta' | Z_1 \text{ is random}]| = |\Pr[X_{M-random}] - \Pr[X_{final}]|.$$

□

E Including Zero as an Identity Component

The identity components for the HIBE schemes that we have discussed so far are all elements of $\mathbb{Z}_p^\times$. Note that in the case of the corresponding IBE schemes, identities are elements of $\mathbb{Z}_p$. For moving to HIBE, it is necessary to remove 0 as a possible identity component. This is because allowing identity components to be zero in the HIBE scheme leads to a trivial attack as explained below.

Attack when 0 is allowed: Suppose that identities are from $\mathbb{Z}_p$ and the attacker $\mathscr{A}$ provides to the challenger, $\widehat{\mathbf{id}}$ as the challenge identity. Then in the second key extraction phase, $\mathscr{A}$ could ask for a key for the identity vector $\widehat{\mathbf{id}} : 0$ and hence essentially obtain a key for $\widehat{\mathbf{id}}$ using which it can decrypt the challenge ciphertext and find out the message that was encrypted. The attack is much stronger than a distinguishing attack. This is not an issue only with our HIBE schemes. The same is true of the BBG-HIBE scheme [BBG05] and the schemes of [LW10, SKOS09]

In the Lewko-Waters paper [LW10], for the case of IBE it is mentioned that the identities are elements of $\mathbb{Z}_p$ (or $\mathbb{Z}_N$ for the composite order pairing based IBE). The description of the HIBE in [LW10] does not explicitly rule out the possibility of having zero as identity component. If identity components for the HIBE in [LW10] are allowed to be 0, then the above mentioned trivial attack applies. So, to get a secure scheme, the identity components of the HIBE in [LW10] should be elements of $\mathbb{Z}_N^*$.

In fact, if one wishes to have zero as identity component in a BBG type scheme, then one has to pay the price of almost doubling the number of public parameters as shown by Chatterjee and Sarkar in [CS07]. Using their technique, we extend one of our schemes (*DPVS-HIBE*) to a scheme that can allow identity components to be zero. We call it *DPVS-HIBE*-0. We also extend this scheme to achieve anonymity to obtain *DPVS-AHIBE*-0.

E.1 *DPVS-HIBE*-0

In this section we present the construction and the proof is given in the following section.

$\mathsf{Setup}\ (\lambda)$: Let h denote the maximum depth of the HIBE. Construct a DPVS $\mathcal{V}_4 = (p, \mathbb{V}, \mathbb{V}^*, \mathbb{G}_T, \bar{e}, \mathbb{A}, \mathbb{A}^*)$ from a Type-3 pairing $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, P_1, P_2)$ whose parameters are generated according to λ. Choose random dual bases $(\mathbf{B}, \mathbf{B}^*) \longleftarrow \mathsf{Dual}(\mathbb{Z}_p^4)$ and $\alpha, \theta_1, \ldots, \theta_h, \eta_1, \ldots, \eta_h \xleftarrow{\mathrm{U}} \mathbb{Z}_p$. Let $N_j = \sum_{i=1}^{j} \eta_j$ for $j = 1, \ldots, h$ denote the partial sums of η_j's. Set the public parameters and master secret as follows.

$\mathcal{PP}$: $(\eta_1 \mathbf{b}_1 P_1, \ldots, \eta_h \mathbf{b}_1 P_1, \theta_1 \mathbf{b}_2 P_1, \ldots, \theta_h \mathbf{b}_2 P_1, e(P_1, P_2)^{N_1 \alpha \mathbf{b}_1 \cdot \mathbf{b}_1^*}, \ldots, e(P_1, P_2)^{N_h \alpha \mathbf{b}_1 \cdot \mathbf{b}_1^*})$.
$\mathcal{MSK}$: $(\alpha \mathbf{b}_1^* P_2, \theta_1 \mathbf{b}_1^* P_2, \ldots, \theta_h \mathbf{b}_1^* P_2, \eta_1 \mathbf{b}_2^* P_2, \ldots, \eta_h \mathbf{b}_2^* P_2)$.

Again, the only quantity of the $\mathcal{MSK}$ which is required to be kept secret by the PKG is $\alpha \mathbf{b}_1^* P_2$.

Encrypt $(\mathcal{PP}, M, \mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell))$: Choose a randomiser s from $\mathbb{Z}_p$. The ciphertext will consist of an element of $\mathbb{G}_T$ and a vector from $\mathbb{V}$ as shown below.

$$C_0 = M \cdot e(P_1, P_2)^{s N_\ell \alpha \mathbf{b}_1 \cdot \mathbf{b}_1^*}, \quad \vec{C}_1 = s \sum_{j=1}^{\ell} (\mathsf{id}_j \theta_j \mathbf{b}_2 P_1 + \eta_j \mathbf{b}_1 P_1).$$

KeyGen $(\mathcal{PP}, \mathcal{MSK}, \mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell))$: Choose $r \xleftarrow{\mathrm{U}} \mathbb{Z}_p$. The secret key for $\mathbf{id}$, $\mathcal{SK}_{\mathbf{id}}$ is given by $\mathcal{SK}_{\mathbf{id}} = (\vec{K}, \vec{D}_{1,\ell+1}, \ldots, \vec{D}_{1,h}, \vec{D}_{2,\ell+1}, \ldots, \vec{D}_{2,h})$ where

$$\vec{K} = \alpha \mathbf{b}_1^* P_2 + r \sum_{j=1}^{\ell} (\mathsf{id}_j \theta_j \mathbf{b}_1^* P_2 - \eta_j \mathbf{b}_2^* P_2),$$

$$\vec{D}_{1,\ell+1} = r \theta_{\ell+1} \mathbf{b}_1^* P_2, \quad \ldots \quad , \vec{D}_{1,h} = r \theta_h \mathbf{b}_1^* P_2.$$
$$\vec{D}_{2,\ell+1} = r \eta_{\ell+1} \mathbf{b}_2^* P_2, \quad \ldots \quad , \vec{D}_{2,h} = r \eta_h \mathbf{b}_2^* P_2.$$

The vector K is used for actual decryption and the remaining vectors are used for delegation.

Delegate $(\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{SK}_{\mathbf{id}}, \mathsf{id}_{\ell+1}, \mathcal{PP})$: Define the notation $\mathbf{id} : \mathsf{id}_{\ell+1} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell, \mathsf{id}_{\ell+1})$, i.e., $\mathbf{id} : \mathsf{id}_{\ell+1}$ denotes the tuple obtained by concatenating $\mathsf{id}_{\ell+1}$ to the end of $\mathbf{id}$.

Pick $\tilde{r} \xleftarrow{\mathrm{U}} \mathbb{Z}_p$. The secret key for the identity $\mathbf{id} : \mathsf{id}_{\ell+1}$ is computed as follows:

$$\vec{K} \leftarrow \vec{K} + \mathsf{id}_{\ell+1} \vec{D}_{1,\ell+1} - \vec{D}_{2,\ell+1} + \tilde{r} \sum_{j=1}^{\ell+1} (\mathsf{id}_j \theta_j \mathbf{b}_1^* P_2 - \eta_j \mathbf{b}_2^* P_2),$$

$$\vec{D}_{1,j} \leftarrow \vec{D}_{1,j} + \tilde{r} \theta_j \mathbf{b}_1^* P_2 \text{ for } \ell + 2 \le j \le h.$$
$$\vec{D}_{2,j} \leftarrow \vec{D}_{2,j} + \tilde{r} \eta_j \mathbf{b}_2^* P_2 \text{ for } \ell + 2 \le j \le h.$$

thus setting the randomiser to $r + \tilde{r}$ for the new secret key given by

$$\mathcal{SK}_{\mathbf{id}:\mathsf{id}_{\ell+1}} = (\vec{K}, \vec{D}_{1,\ell+2}, \ldots, \vec{D}_{1,h}, \vec{D}_{2,\ell+2}, \ldots, \vec{D}_{2,h}).$$

Decrypt $(\mathcal{C}, \mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell), \mathcal{SK}_{\mathbf{id}}, \mathcal{PP})$: Decryption is done as follows.

$$M = C_0 / \bar{e}(\vec{C}_1, \vec{K}).$$

Correctness of decryption. The following computation shows the correctness of decryption.

$$\begin{aligned}
\bar{e}(\vec{C}_1, \vec{K}) &= \bar{e}\left(s\sum_{j=1}^{\ell}(\mathsf{id}_j\theta_j\mathbf{b}_2P_1 + \eta_j\mathbf{b}_1P_1),\ \alpha\mathbf{b}_1^*P_2 + r\sum_{j=1}^{\ell}(\mathsf{id}_j\theta_j\mathbf{b}_1^* - \eta_j\mathbf{b}_2^*)P_2 \right) \\
&= \bar{e}\left(\mathbf{b}_1P_1, \mathbf{b}_1^*P_2\right)^{N_\ell s\alpha} \cdot \bar{e}\left(\sum_{j=1}^{\ell}\eta_j\mathbf{b}_1P_1, \sum_{j=1}^{\ell}\mathsf{id}_j\theta_j\mathbf{b}_1^*P_2\right)^{rs} \cdot \bar{e}\left(\sum_{j=1}^{\ell}\mathsf{id}_j\theta_j\mathbf{b}_2P_1,\ \sum_{j=1}^{\ell}\eta_j\mathbf{b}_2^*P_2\right)^{-rs} \\
&= e\left(P_1,\ P_2\right)^{sN_\ell\alpha\mathbf{b}_1\cdot\mathbf{b}_1^*} \cdot e\left(P_1,\ P_2\right)^{rs(\mathbf{b}_1\cdot\mathbf{b}_1^*)\sum_{i,j=1}^{\ell}(\eta_i\mathsf{id}_j\theta_j)} \cdot e\left(P_1,\ P_2\right)^{-rs(\mathbf{b}_2\cdot\mathbf{b}_2^*)\sum_{i,j=1}^{\ell}(\eta_i\mathsf{id}_j\theta_j)} \\
&= e\left(P_1,\ P_2\right)^{sN_\ell\alpha\mathbf{b}_1\cdot\mathbf{b}_1^*}
\end{aligned}$$

E.2 Security Proof for *DPVS-HIBE*-0

The security arguments are more or less same as that of *DPVS-HIBE*. We need to first define semi-functional ciphertext and keys.

Semi-functional ciphertext: Let $(C_0, \vec{C}_1)$ be a normal ciphertext for some message M and identity vector $\mathbf{id}$. Semi-functional ciphertext is obtained by modifying $\vec{C}_1$ as follows:

$$t_1, t_2 \xleftarrow{\mathrm{U}} \mathbb{Z}_p,$$

$$\vec{C}_1 \leftarrow \vec{C}_1 + (t_1\mathbf{b}_3 + t_2\mathbf{b}_4)P_1.$$

Semi-functional key: For an identity vector $\mathbf{id}$ of length ℓ, let $(\vec{K}, \vec{D}_{\ell+1}, \ldots, \vec{D}_h)$ be a normal key. A semi-functional key for $\mathbf{id}$ is computed as follows:

$$z_1, z_2, (x_{1,j}), (x_{2,j}), (y_{1,j}), (y_{2,j}) \xleftarrow{\mathrm{U}} \mathbb{Z}_p \text{ where } j = \ell+1, \ldots, h$$

$$\vec{K} \leftarrow \vec{K} + (z_1\mathbf{b}_3^* + z_2\mathbf{b}_4^*)P_2$$

$$\vec{D}_{1,j} \leftarrow \vec{D}_{1,j} + x_{1,j}\mathbf{b}_3^*P_2 + y_{1,j}\mathbf{b}_4^*P_2 \quad \text{for } \ell+1 \le j \le h$$

$$\vec{D}_{2,j} \leftarrow \vec{D}_{2,j} + x_{2,j}\mathbf{b}_3^*P_2 + y_{2,j}\mathbf{b}_4^*P_2 \quad \text{for } \ell+1 \le j \le h$$

The hybrid argument is over a sequence of $q+4$ games, all with the usual definition. We use the notation $\mathcal{H}_\ell(\mathbf{id}) = \sum_{j=1}^{\ell}(\mathsf{id}_j\theta_j)$ where $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$. Let $\gamma_p = \left(1 - \frac{1}{p}\right)$. The definition of $\mathrm{Game}_{restricted}$ is different here - the simulator makes sure that $N_{\widehat{\ell}}\mathcal{H}_\ell(\mathbf{id}_k) \not\equiv N_\ell\mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \pmod p$. The probability of such an event, however, remains the same as mentioned in the lemma below.

Lemma E.1. $|\Pr[X_{actual}] - \Pr[X_{restricted}| \le q/p$.

Lemma E.2. *If there exists a adversary $\mathscr{A}$ against DPVS-HIBE making at most q key extraction queries, then we can build an algorithm $\mathscr{B}$ having advantage at least $\gamma_p^2|\Pr[X_{restricted}] - \Pr[X_0]|$ in solving the* DS1 *problem with $n = 4, k = 2$.*

Proof. The algorithm $\mathscr{B}$ receives $(\mathcal{G}, \mathbf{d}_1 P_1, \ldots, \mathbf{d}_4 P_1, \mathbf{d}_1^* P_2, \mathbf{d}_2^* P_2, \vec{U}_1, \vec{U}_2, \mu_2, \vec{T}_1, \vec{T}_2)$ as an instance of DS1 problem. Here $\vec{U}_1 = (\mu_1 \mathbf{d}_1^* + \mu_2 \mathbf{d}_3^*) P_2$ and $\vec{U}_2 = (\mu_1 \mathbf{d}_2^* + \mu_2 \mathbf{d}_4^*) P_2$. So $\mathscr{B}$'s task is to decide whether $\vec{T}_1, \vec{T}_2$ are real or random and accordingly return a bit 0 or 1, respectively.

Setup: $\mathscr{B}$ first chooses $\mathbf{X} \xleftarrow{\mathrm{U}} \mathbb{Z}_p^{2\times 2}$ and if $\mathbf{X} \notin GL(2, \mathbb{Z}_p)$, aborts the game and returns a random bit. Otherwise, it sets the dual bases $(\mathbf{B}, \mathbf{B}^*)$ of the HIBE scheme as follows.

$$\mathbf{b}_1 = \mathbf{d}_1, \;\; \mathbf{b}_2 = \mathbf{d}_2, \;\; (\mathbf{b}_3, \mathbf{b}_4) = (\mathbf{d}_3, \mathbf{d}_4)\mathbf{X},$$

$$\mathbf{b}_1^* = \mathbf{d}_1^*, \;\; \mathbf{b}_2 = \mathbf{d}_2^*, \;\; (\mathbf{b}_3^*, \mathbf{b}_4^*) = (\mathbf{d}_3^*, \mathbf{d}_4^*)(\mathbf{X}^{-1})^T.$$

It then chooses $\alpha, (\theta_j), (\eta_j) \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ and computes the partial sums (N_j). The following public parameters are provided to $\mathscr{A}$.

$$\mathcal{PP} = (\mathcal{G}, \mathbf{b}_1 P_1, (\theta_j \mathbf{b}_2 P_1), (\eta_j \mathbf{b}_1 P_1), (e(P_1, P_2)^{N_j \alpha \mathbf{b}_1 \cdot \mathbf{b}_1^*}))$$

$\mathscr{B}$ knows the master secret key $\mathcal{MSK} = (\alpha \mathbf{b}_1^* P_2, (\theta_j \mathbf{b}_1^* P_2), (\eta_j \mathbf{b}_2^* P_2))$.

Phase 1: $\mathscr{A}$ makes a number of key extract queries. $\mathscr{B}$ returns a normal key generated using the KeyGen algorithm for every key extract query made by $\mathscr{A}$.

Challenge: $\mathscr{B}$ receives the two messages M_0, M_1 and challenge identity $\widehat{\mathbf{id}} = (\widehat{\mathsf{id}}_1, \ldots, \widehat{\mathsf{id}}_{\widehat{\ell}})$ from $\mathscr{A}$. It chooses $\beta \xleftarrow{\mathrm{U}} \{0, 1\}$ and encrypts M_β to $\widehat{\mathbf{id}}$ as:

$$C_0 = M_\beta \cdot \bar{e}(\vec{T}_1, \mathbf{b}_1^* P_2)^{N_{\widehat{\ell}} \alpha}, \qquad \vec{C}_1 = \sum_{j=1}^{\widehat{\ell}} \left(\widehat{\mathsf{id}}_j \theta_j \vec{T}_2 - \eta_j \vec{T}_1 \right),$$

implicitly setting $s = \tau_1$. Ciphertext $\widehat{\mathcal{C}} = (C_0, \vec{C}_1)$ is returned to $\mathscr{A}$.

Now if $\vec{T}_1, \vec{T}_2$ are real, $\widehat{\mathcal{C}}$ is distributed as a normal ciphertext and in this case $\mathscr{B}$ simulates $\mathrm{Game}_{restricted}$. Otherwise, $\widehat{\mathcal{C}}$ will be semi-functional and $\mathscr{B}$ simulates Game_0. The computation shown below illustrates this fact. Suppose that $\vec{T}_1 = \tau_1 \mathbf{d}_1 P_1$ and $\vec{T}_2 = \tau_1 \mathbf{d}_2 P_1$. Then

$$\begin{aligned} \vec{C}_1 &= \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \vec{T}_2 - N_{\widehat{\ell}} \vec{T}_1 \\ &= \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \tau_1 \mathbf{d}_2 P_1 + N_{\widehat{\ell}} \tau_1 \mathbf{d}_1 P_1 \\ &= s \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \mathbf{b}_2 P_1 + s N_{\widehat{\ell}} \mathbf{b}_1 P_1. \end{aligned}$$

Next consider the case when $\vec{T}_1 = (\tau_1 \mathbf{d}_1 + \tau_2 \mathbf{d}_3) P_1$ and $\vec{T}_2 = (\tau_1 \mathbf{d}_2 + \tau_2 \mathbf{d}_4) P_1$. We have

$$\begin{aligned} \vec{C}_1 &= N_{\widehat{\ell}} \vec{T}_1 + \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \vec{T}_2 \\ &= N_{\widehat{\ell}} (\tau_1 \mathbf{d}_1 + \tau_2 \mathbf{d}_3) P_1 + \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) (\tau_1 \mathbf{d}_2 + \tau_2 \mathbf{d}_4) P_1 \\ &= N_{\widehat{\ell}} \tau_1 \mathbf{d}_1 P_1 + \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \tau_1 \mathbf{d}_2 P_1 + N_{\widehat{\ell}} \tau_2 \mathbf{d}_3 P_1 + \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \tau_2 \mathbf{d}_4 P_1 \\ &= s N_{\widehat{\ell}} \mathbf{b}_1 P_1 + s \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \mathbf{b}_2 P_1 + (\mathbf{d}_3, \mathbf{d}_4)(N_{\widehat{\ell}}, \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}))^T \tau_2 P_1 \\ &= s N_{\widehat{\ell}} \mathbf{b}_1 P_1 + s \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \mathbf{b}_2 P_1 + (\mathbf{b}_3, \mathbf{b}_4) \mathbf{X}^{-1} (N_{\widehat{\ell}}, \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}))^T \tau_2 P_1, \end{aligned}$$

thus setting $(t_1, t_2) = \tau_2 \mathbf{X}^{-1} (N_{\widehat{\ell}}, \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}))^T$ which has the appropriate distribution due to the choice of $\mathbf{X}$ and the fact that all information provided to $\mathscr{A}$ are independent of $\mathbf{X}$. $\mathscr{B}$'s simulation is, therefore, perfect.

Phase 2: As in first phase, $\mathscr{B}$ returns a normal key for every query.

Guess: $\mathscr{A}$ returns its guess β'.

If the adversary wins the game then $\mathscr{B}$ returns 1; otherwise it returns 0. Let Y_{real} and Y_{random} denote the events that $\mathscr{B}$ returns 1 when $\vec{T}_1, \vec{T}_2$ are real and random respectively. Let the event abort denote the event that $\mathscr{B}$ aborts the game. Then, we have

$$\begin{aligned}\Pr[Y_{real}] &= \Pr[Y_{real}|\mathsf{abort}]\Pr[\mathsf{abort}] + \Pr[Y_{real}|\overline{\mathsf{abort}}]\Pr[\overline{\mathsf{abort}}] \\ &= \frac{1}{2}\Pr[\mathsf{abort}] + \Pr[X_{restricted}]\Pr[\overline{\mathsf{abort}}], \qquad (7) \\ \Pr[Y_{random}] &= \Pr[Y_{random}|\mathsf{abort}]\Pr[\mathsf{abort}] + \Pr[Y_{random}|\overline{\mathsf{abort}}]\Pr[\overline{\mathsf{abort}}] \\ &= \frac{1}{2}\Pr[\mathsf{abort}] + \Pr[X_0]\Pr[\overline{\mathsf{abort}}] \qquad (8)\end{aligned}$$

Recall that $\mathscr{B}$ aborts when $\mathbf{X}$ is not invertible. From B.2 it follows that $\Pr[\overline{\mathsf{abort}}] = (p^2-1)(p-1)/p^3 \geq (p-1)^2/p^2 = \gamma_p^2$. Subtracting (8) from (7), we obtain

$$\begin{aligned}\Pr[Y_{real}] - \Pr[Y_{random}] &= (\Pr[X_{restricted}] - \Pr[X_0])\Pr[\overline{\mathsf{abort}}] \\ &\geq (\Pr[X_{restricted}] - \Pr[X_0])\gamma_p^2.\end{aligned}$$

$\mathscr{B}$'s advantage in solving the DS1 problem is given by

$$\mathsf{Adv}_{\mathcal{V}_4}^{\mathrm{DS1}}(\mathscr{B}) = |\Pr[Y_{real}] - \Pr[Y_{random}]| \geq |\Pr[X_{restricted}] - \Pr[X_0]|\gamma_p^2.$$

□

Lemma E.3. *If there exists an adversary $\mathscr{A}$ against DPVS-HIBE making at most q key extraction queries, then we can build an algorithm $\mathscr{B}$ having advantage at least $\gamma_p^2\,|\Pr[X_{k-1}] - \Pr[X_k]|$ in solving the* DS2 *problem with $n = 4, k = 2$.*

Proof. An instance $(\mathcal{G}, \mathbf{d}_1 P_1, \mathbf{d}_2 P_1, \mathbf{d}_1^* P_2, \ldots, \mathbf{d}_4^* P_2, \vec{U}_1, \vec{U}_2, \mu_2, \vec{T}_1, \vec{T}_2)$ of DS2 is provided to $\mathscr{B}$. Here $\vec{U}_1 = (\mu_1\mathbf{d}_1 + \mu_2\mathbf{d}_3)P_1$ and $\vec{U}_2 = (\mu_1\mathbf{d}_2 + \mu_2\mathbf{d}_4)P_1$. $\mathscr{B}$ has to determine whether $\vec{T}_1, \vec{T}_2$ are real or random - i.e., whether they are distributed as $\tau_1\mathbf{d}_1^*P_2, \tau_1\mathbf{d}_2^*P_2$ or $(\tau_1\mathbf{d}_1^* + \tau_2\mathbf{d}_3^*)P_2, (\tau_1\mathbf{d}_2^* + \tau_2\mathbf{d}_4^*)P_2$.

Setup: $\mathscr{B}$ samples $\mathbf{X} = (\chi_{\iota,\jmath}) \xleftarrow{\mathrm{U}} \mathbb{Z}_p^{2\times 2}$. If $\mathbf{X}$ is not invertible, it aborts and returns a random bit; otherwise it proceeds to set the dual bases $(\mathbf{B}, \mathbf{B}^*)$ of the HIBE scheme as follows.

$$\mathbf{b}_1 = \mathbf{d}_1,\ \ \mathbf{b}_2 = \mathbf{d}_2,\ \ (\mathbf{b}_3, \mathbf{b}_4) = (\mathbf{d}_3, \mathbf{d}_4)\mathbf{X},$$

$$\mathbf{b}_1^* = \mathbf{d}_1^*,\ \ \mathbf{b}_2^* = \mathbf{d}_2^*,\ \ (\mathbf{b}_3^*, \mathbf{b}_4^*) = (\mathbf{d}_3^*, \mathbf{d}_4^*)(\mathbf{X}^{-1})^T.$$

$\mathscr{B}$ then picks $\alpha, \theta_1, \ldots, \theta_h \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ and sends the public parameters $\mathcal{PP}$ to $\mathscr{A}$. $\mathscr{B}$ knows $\mathcal{MSK}$ and hence can create normal keys. Knowledge of $\mathbf{b}_3^*, \mathbf{b}_4^*$ enables it to generate semi-functional keys too.

Phases 1 and 2: Let $\mathbf{id}_1, \mathbf{id}_2, \ldots, \mathbf{id}_q$ denote the identities for which $\mathscr{A}$ requests the secret keys. For $i > k$, $\mathscr{B}$ generates a normal key for $\mathbf{id}_i$ using the master secret and returns the resulting key to $\mathscr{A}$. For $i < k$, $\mathscr{B}$ first generates a normal key and then modifies it appropriately to obtain a semi-functional key. The resulting key is returned to $\mathscr{A}$. In case of the k-th query (i.e., $i = k$), $\mathscr{B}$ generates the key as follows. Let $\mathbf{id}_k = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$.

$$\vec{K} = \alpha\mathbf{b}_1^* P_2 + \mathcal{H}_\ell(\mathbf{id}_k)\vec{T}_1 - N_\ell\vec{T}_2,$$

$$\vec{D}_{1,\ell+1} = \theta_{\ell+1}\vec{T}_1,\ \ \ldots,\ \ \vec{D}_{1,h} = \theta_h\vec{T}_1.$$

$$\vec{D}_{2,\ell+1} = \eta_{\ell+1}\vec{T}_2,\ \ \ldots,\ \ \vec{D}_{2,h} = \eta_h\vec{T}_2.$$

The key $\mathcal{SK}_{\mathbf{id}_k} = (\vec{K}, (\vec{D}_{1,j})_{j=\ell+1}^{h}, (\vec{D}_{2,j})_{j=\ell+1}^{h})$ is returned to $\mathscr{A}$.

If $\vec{T}_1, \vec{T}_2$ are real, then it is clear that $\mathcal{SK}_{\mathbf{id}_k}$ is going to be normal with r being set to τ_1 implicitly. In this case $\mathscr{B}$ simulates Game_{k-1}. Otherwise $\vec{T}_1, \vec{T}_2$ would be random and the key will be semi-functional as justified below.

$$\begin{aligned}
\vec{K} &= \alpha \mathbf{b}_1^* P_2 + \mathcal{H}_\ell(\mathbf{id}_k)\vec{T}_1 - N_\ell \vec{T}_2 \\
&= \alpha \mathbf{b}_1^* P_2 + \mathcal{H}_\ell(\mathbf{id}_k)\tau_1 \mathbf{d}_1^* P_2 - N_\ell \tau_1 \mathbf{d}_2^* P_2 + \tau_2 \mathcal{H}_\ell(\mathbf{id}_k)\mathbf{d}_3^* P_2 - N_\ell \tau_2 \mathbf{d}_4^* P_2 \\
&= \alpha \mathbf{b}_1^* P_2 + r\mathcal{H}_\ell(\mathbf{id}_k)\mathbf{b}_1^* P_2 - rN_\ell \mathbf{b}_2^* P_2 + (\mathbf{b}_3^*, \mathbf{b}_4^*)\mathbf{X}^T \tau_2 (\mathcal{H}_\ell(\mathbf{id}_k), -N_\ell)^T P_2, \\
\vec{D}_{1,j} &= \theta_j \vec{T}_1 \\
&= \theta_j(\tau_1 \mathbf{d}_1^* + \tau_2 \mathbf{d}_3^*) P_2 \\
&= \theta_j \tau_1 \mathbf{b}_1^* P_2 + \theta_j \tau_2 (\chi_{1,1}\mathbf{b}_3^* + \chi_{1,2}\mathbf{b}_4^*) P_2, \\
\vec{D}_{2,j} &= \eta_j \vec{T}_2 \\
&= \eta_j(\tau_1 \mathbf{d}_2^* + \tau_2 \mathbf{d}_4^*) P_2 \\
&= \eta_j \tau_1 \mathbf{b}_2^* P_2 + \eta_j \tau_2 (\chi_{2,1}\mathbf{b}_3^* + \chi_{2,2}\mathbf{b}_4^*) P_2,
\end{aligned}$$

setting $r = \tau_1$, $(z_1, z_2)^T = \tau_2 \mathbf{X}^T (\mathcal{H}_\ell(\mathbf{id}_k), -N_\ell)^T$, $x_{1,j} = \theta_j \tau_2 \chi_{1,1}$, $y_{1,j} = \theta_j \tau_2 \chi_{1,2}$, $x_{2,j} = \eta_j \tau_2 \chi_{2,1}$ and $y_{2,j} = \eta_j \tau_2 \chi_{2,2}$ for $\ell + 1 \le j \le h$. Consequently, $\mathscr{B}$ simulates Game_k.

Challenge: $\mathscr{B}$ receives messages M_0, M_1 and challenge identity $\widehat{\mathbf{id}} = (\widehat{\mathsf{id}}_1, \ldots, \widehat{\mathsf{id}}_{\widehat{\ell}})$ from $\mathscr{A}$. It chooses $\beta \in \{0, 1\}$ at random. $\mathscr{B}$ encrypts M_β to $\widehat{\mathbf{id}}$ as

$$C_0 = M_\beta \cdot \bar{e}(\vec{U}_1, \mathbf{b}_1^* P_2)^{N_{\widehat{\ell}}\alpha}, \;\; \vec{C}_1 = N_{\widehat{\ell}} \vec{U}_1 + \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})\vec{U}_2$$

and returns $\widehat{\mathcal{C}} = (C_0, \vec{C}_1)$ to $\mathscr{A}$. Component $\vec{C}_1$ is properly distributed since

$$\begin{aligned}
\vec{C}_1 &= N_{\widehat{\ell}} \vec{U}_1 + \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})\vec{U}_2 \\
&= N_{\widehat{\ell}} \mu_1 \mathbf{d}_1 P_1 + \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})\mu_1 \mathbf{d}_2 P_1 + N_{\widehat{\ell}} \mu_2 \mathbf{d}_3 P_1 + \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})\mu_2 \mathbf{d}_4 P_1 \\
&= N_{\widehat{\ell}} \mu_1 \mathbf{b}_1 P_1 + \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})\mu_1 \mathbf{b}_2 P_1 + (\mathbf{b}_3, \mathbf{b}_4)\mathbf{X}^{-1} \mu_2 (N_{\widehat{\ell}}, \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}))^T P_1
\end{aligned}$$

implicitly setting $s = \mu_1$ and $(t_1, t_2)^T = \mu_2 \mathbf{X}^{-1} (N_{\widehat{\ell}}, \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}))^T$.

Guess: $\mathscr{A}$ returns its guess β' of β.

We first argue that all information provided to $\mathscr{A}$ have the right distribution. This is straightforward in case of Game_{k-1}. Due to the choice of the θ_j's and $\mathbf{X}$, the scalars $x_{1,j}$, $y_{1,j}$, $x_{2,j}$, $y_{2,j}$ (for $\ell + 1 \le j \le h$) will be uniformly distributed in $\mathbb{Z}_p$. Therefore the semi-functional components of $\vec{D}_{i,j}$'s will be properly distributed. Next we show that in Game_k, the semi-functional components of $\mathcal{SK}_{\mathbf{id}_k}$ and $\widehat{\mathcal{C}}$ are properly distributed in $\mathscr{A}$'s view. For this, we need only argue that the co-efficient vectors (t_1, t_2) and (z_1, z_2) given by

$$\begin{aligned}
(t_1, t_2)^T &= \mu_2 \mathbf{X}^{-1} (N_{\widehat{\ell}}, \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}))^T \text{ and} \\
(z_1, z_2)^T &= \tau_2 \mathbf{X}^T (\mathcal{H}_\ell(\mathbf{id}_k), -N_\ell)^T
\end{aligned}$$

are uniformly and independently distributed. Note that all the information provided to the adversary except for the challenge ciphertext and the response to the k-th key extraction query are independent of the matrix $\mathbf{X}$. Also, observe that $\mathbf{id}_k \neq \widehat{\mathbf{id}}$ and the quantities θ_j, η_j are uniformly distributed over $\mathbb{Z}_p$.

Hence, by Lemma B.1 it follows that the coefficient vectors are uniformly and independently distributed except when $N_{\widehat{\ell}}\mathcal{H}_\ell(\mathbf{id}_k) \equiv N_\ell \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \pmod p$.

We now obtain a bound $\mathsf{Adv}^{\mathrm{DS2}}_{\mathcal{G}}(\mathscr{B})$. If $\beta = \beta'$, the $\mathscr{B}$ returns 1; otherwise it returns 0. Let Y_{real} and Y_{random} denote the events that $\mathscr{B}$ returns 1 when $\vec{T}_1, \vec{T}_2$ are real and random respectively; let abort denote the event that $\mathscr{B}$ aborts. We have

$$\begin{aligned}
\Pr[Y_{real}] &= \Pr[Y_{real}|\mathsf{abort}]\Pr[\mathsf{abort}] + \Pr[Y_{real}|\overline{\mathsf{abort}}]\Pr[\overline{\mathsf{abort}}] \\
&= \frac{1}{2}\Pr[\mathsf{abort}] + \Pr[X_{k-1}]\Pr[\overline{\mathsf{abort}}], \qquad (9) \\
\Pr[Y_{random}] &= \Pr[Y_{random}|\mathsf{abort}]\Pr[\mathsf{abort}] + \Pr[Y_{random}|\overline{\mathsf{abort}}]\Pr[\overline{\mathsf{abort}}] \\
&= \frac{1}{2}\Pr[\mathsf{abort}] + \Pr[X_k]\Pr[\overline{\mathsf{abort}}] \qquad (10)
\end{aligned}$$

Subtracting (9) from (10) gives us

$$\begin{aligned}
\Pr[Y_{real}] - \Pr[Y_{random}] &= \left(\Pr[X_{k-1}] - \left(1 - \frac{1}{p}\right)\Pr[X_k]\right)\Pr[\overline{\mathsf{abort}}] \\
&\geq \left(\Pr[X_{k-1}] - \Pr[X_k]\right)\gamma_p^2
\end{aligned}$$

and hence

$$\mathsf{Adv}^{\mathrm{DS2}}_{\mathcal{V}_4}(\mathscr{B}) = \left|\Pr[Y_{real}] - \Pr[Y_{random}]\right| \geq \gamma_p^2 \left|\Pr[X_{k-1}] - \Pr[X_k]\right|.$$

□

Lemma E.4. *Suppose an adversary $\mathscr{A}$ against $\mathcal{DPVS}$-$\mathcal{HIBE}$ makes at most q key extraction queries, then one can construct an algorithms $\mathscr{B}$ which has advantage $\left|\Pr[X_q] - \Pr[X_{final}]\right|$ in breaking the* DS1 *with $n = 4$ and $k = 2$.*

Proof. $\mathscr{B}$ receives an instance $(\mathcal{G}, \mathbf{d}_1 P_1, \ldots, \mathbf{d}_4 P_1, \mathbf{d}_1^* P_2, \mathbf{d}_2^* P_2, \vec{U}_1, \vec{U}_2, \mu_2, \vec{T}_1, \vec{T}_2)$ of DS1 problem and has to decide whether $\vec{T}_1, \vec{T}_2$ are distributed as $\tau_1\mathbf{d}_1 P_1, \tau_1 \mathbf{d}_2 P_1$ or $(\tau_1\mathbf{d}_1 + \tau_2\mathbf{d}_3)P_1, (\tau_1\mathbf{d}_2 + \tau_2\mathbf{d}_4)P_1$ (whether they are real or random respectively) and accordingly return a bit 0 or 1. $\mathscr{B}$ will simulate each phase of the security game as described below.

Setup: $\mathscr{B}$ constructs the dual bases $\mathbf{B}, \mathbf{B}^*$ as follows.

$$(\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3, \mathbf{b}_4) = (\mathbf{d}_3, \mathbf{d}_4, \mathbf{d}_1, \mathbf{d}_2),$$

$$(\mathbf{b}_1^*, \mathbf{b}_2^*, \mathbf{b}_3^*, \mathbf{b}_4^*) = (\mathbf{d}_3^*, \mathbf{d}_4^*, \mathbf{d}_1^*, \mathbf{d}_2^*).$$

Note that $(\mathbf{B}, \mathbf{B}^*)$ are obtained by just a cyclic shift of the vectors in $(\mathbf{D}, \mathbf{D}^*)$. Then $\mathscr{B}$ chooses $\alpha', (\theta_j), (\eta_j) \xleftarrow{\mathrm{U}} \mathbb{Z}_p$, computes

$$\begin{aligned}
\bar{e}(\mathbf{d}_3 P_1, \vec{U}_1)^{N_j \alpha'} &= \bar{e}\left(\mathbf{d}_3 P_1, (\mu_1 \mathbf{d}_1^* + \mu_2 \mathbf{d}_3^*)P_2\right)^{N_j \alpha'} \\
&= \bar{e}\left(\mathbf{d}_3 P_1, \mathbf{d}_3^* P_2\right)^{\mu_2 N_j \alpha'} \\
&= e(P_1, P_2)^{\alpha N_j \mathbf{b}_1 \cdot \mathbf{b}_1^*},
\end{aligned}$$

for all $j \in \{1, \ldots, h\}$, setting $\alpha = \alpha' \mu_2$, and provides the following public parameters to $\mathscr{A}$.

$$\mathcal{PP} = (\mathcal{G}, (\eta_j \mathbf{b}_1 P_1), (\theta_j \mathbf{b}_2 P_1), (e(P_1, P_2)^{\alpha N_j \mathbf{b}_1 \cdot \mathbf{b}_1^*}))$$

Note that $\alpha \mathbf{b}_1^* P_2$ is not known to $\mathscr{B}$ and hence it cannot create normal keys.

Key Extraction Phases 1 and 2: Let $\mathbf{id}_1, \ldots, \mathbf{id}_q$ be the identities provided by $\mathscr{A}$ for key extraction. The response to the k-th query $\mathbf{id}_k = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$, $\mathcal{SK}_{\mathbf{id}_k}$, is computed as follows. $\mathscr{B}$ picks $r', z_1', z_2', (x_{1,j}', x_{2,j}, y_{1,j}, y_{2,j}')_{j=\ell+1}^h \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ and computes

$$\vec{K} = (\alpha' + r' \mathcal{H}_\ell(\mathbf{id}_k))\vec{U}_1 - r' N_\ell \vec{U}_2 + z_1' \mathbf{d}_1^* P_2 + z_2' \mathbf{d}_2^* P_2,$$
$$\vec{D}_{1,j} = r' \theta_j \vec{U}_1 + x_{1,j}' \mathbf{d}_1^* P_2 + y_{1,j} \mathbf{d}_2^* P_2 \text{ for } \ell + 1 \le j \le h,$$
$$\vec{D}_{2,j} = r' \eta_j \vec{U}_2 + x_{2,j} \mathbf{d}_1^* P_2 + y_{2,j}' \mathbf{d}_2^* P_2 \text{ for } \ell + 1 \le j \le h,$$

implicitly setting $r = r'\mu_2$, $z_1 = z_1' + \mu_1(\alpha' + r' \mathcal{H}_\ell(\mathbf{id}_k))$, $z_2 = z_2' - r'\mu_1$, $x_{1,j} = x_{1,j}' + r'\theta_j \mu_1$ and $y_{2,j} = y_{2,j}' + r' \eta_j \mu_1$ for $j = \ell + 1, \ldots, h$. The following computation shows that the key $\mathcal{SK}_{\mathbf{id}_k} = (\vec{K}, (\vec{D}_{1,j}, \vec{D}_{2,j})_{j=\ell+1}^h)$ is a properly distributed semi-functional key for $\mathbf{id}_k$.

$$\begin{aligned}
\vec{K} &= (\alpha' + r' \mathcal{H}_\ell(\mathbf{id}_k))\vec{U}_1 - r' N_\ell \vec{U}_2 + z_1' \mathbf{d}_1^* P_2 + z_2' \mathbf{d}_2^* P_2 \\
&= (\alpha' + r' \mathcal{H}_\ell(\mathbf{id}_k))(\mu_1 \mathbf{d}_1^* + \mu_2 \mathbf{d}_3^*) P_2 - r' N_\ell (\mu_1 \mathbf{d}_2^* + \mu_2 \mathbf{d}_4^*) P_2 + z_1' \mathbf{d}_1^* P_2 + z_2' \mathbf{d}_2^* P_2 \\
&= (\mu_2 \alpha' + \mu_2 r' \mathcal{H}_\ell(\mathbf{id}_k)) \mathbf{d}_3^* P_2 - r' N_\ell \mu_2 \mathbf{d}_4^* P_2 + (z_1' + (\alpha' + r' \mathcal{H}_\ell(\mathbf{id}_k))\mu_1) \mathbf{d}_1^* P_2 + (z_2' - r' N_\ell \mu_1) \mathbf{d}_2^* P_2 \\
&= (\alpha + r \mathcal{H}_\ell(\mathbf{id}_k)) \mathbf{b}_1^* P_2 - r N_\ell \mathbf{b}_2^* P_2 + z_1 \mathbf{b}_3^* P_2 + z_2 \mathbf{b}_4^* P_2
\end{aligned}$$

$$\begin{aligned}
\vec{D}_{1,j} &= r' \theta_j \vec{U}_1 + x_{1,j}' \mathbf{d}_1^* P_2 + y_{1,j} \mathbf{d}_2^* P_2 \\
&= r' \theta_j (\mu_1 \mathbf{d}_1^* + \mu_2 \mathbf{d}_3^*) P_2 + x_{1,j}' \mathbf{d}_1^* P_2 + y_{1,j} \mathbf{d}_2^* P_2 \\
&= r' \theta_j \mu_2 \mathbf{d}_3^* P_2 + (x_{1,j}' + r' \theta_j \mu_1) \mathbf{d}_1^* P_2 + y_{1,j} \mathbf{d}_2^* P_2 \\
&= r \theta_j \mathbf{b}_1^* P_2 + x_{1,j} \mathbf{b}_3^* P_2 + y_{1,j} \mathbf{b}_4^* P_2 \\
\vec{D}_{2,j} &= r' \eta_j \vec{U}_2 + x_{2,j} \mathbf{d}_1^* P_2 + y_{2,j}' \mathbf{d}_2^* P_2 \\
&= r' \eta_j (\mu_1 \mathbf{d}_2^* + \mu_2 \mathbf{d}_4^*) P_2 + x_{2,j} \mathbf{d}_1^* P_2 + y_{2,j}' \mathbf{d}_2^* P_2 \\
&= r' \eta_j \mu_2 \mathbf{d}_4^* P_2 + x_{2,j} \mathbf{d}_1^* P_2 + (y_{1,j} + r' \eta_j \mu_1) \mathbf{d}_2^* P_2 \\
&= r \eta_j \mathbf{b}_2^* P_2 + x_{2,j} \mathbf{b}_3^* P_2 + y_{2,j} \mathbf{b}_4^* P_2
\end{aligned}$$

Challenge: $\mathscr{A}$ sends two messages M_0, M_1 and the challenge identity vector $\widehat{\mathbf{id}} = (\widehat{\mathsf{id}}_1, \ldots, \widehat{\mathsf{id}}_{\widehat{\ell}})$. The simulator now chooses $\beta \xleftarrow{\mathrm{U}} \{0,1\}$, $s', t_1', t_2' \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ and computes the challenge ciphertext as $\widehat{\mathcal{C}} = (C_0, \vec{C}_1)$ as follows.

$$C_0 = M_\beta \cdot \bar{e}(\mathbf{d}_3 P_1, \vec{U}_1)^{N_{\widehat{\ell}} \alpha' s'},$$
$$\vec{C}_1 = s' N_{\widehat{\ell}} \mathbf{d}_3 P_1 + s' \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \mathbf{d}_4 P_1 + N_{\widehat{\ell}} \vec{T}_1 + \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \vec{T}_2 + t_1' \mathbf{d}_1 + t_2' \mathbf{d}_2.$$

If $\vec{T}_1$,$\vec{T}_2$ are real then

$$\begin{aligned}
\vec{C}_1 &= s' N_{\widehat{\ell}} \mathbf{d}_3 P_1 + s' \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \mathbf{d}_4 P_1 + N_{\widehat{\ell}} \tau_1 \mathbf{d}_1 P_1 + \tau_1 \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \mathbf{d}_2 P_1 + t_1' \mathbf{d}_1 + t_2' \mathbf{d}_2 \\
&= s' N_{\widehat{\ell}} \mathbf{b}_1 P_1 + s' \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \mathbf{b}_2 P_1 + (t_1' + N_{\widehat{\ell}} \tau_1) \mathbf{b}_3 P_1 + (t_2' + \tau_1 \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})) \mathbf{b}_4 P_1,
\end{aligned}$$

making $\widehat{\mathcal{C}}$ a semi-functional encryption of M_β with $s = s'$, $t_1 = t_1' + N_{\widehat{\ell}} \tau_1$ and $t_2 = t_2' + \tau_1 \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})$. Thus $\mathscr{B}$ simulates Game_q. Otherwise, if $\vec{T}_1$,$\vec{T}_2$ are random, $\widehat{\mathcal{C}}$ will be a semi-functional encryption of a random message and as a result $\mathscr{B}$ simulates Game_{final}. This is because C_1 is given by

$$\begin{aligned}
\vec{C}_1 &= (s' + \tau_2) N_{\widehat{\ell}} \mathbf{d}_3 P_1 + (s' + \tau_2) \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \mathbf{d}_4 P_1 + (t_1' + N_{\widehat{\ell}} \tau_1) \mathbf{d}_1 P_1 + (t_2' + \tau_1 \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})) \mathbf{d}_2 P_1 \\
&= (s' + \tau_2) N_{\widehat{\ell}} \mathbf{b}_1 P_1 + (s' + \tau_2) \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}}) \mathbf{b}_2 P_1 + (t_1' + N_{\widehat{\ell}} \tau_1) \mathbf{b}_3 P_1 + (t_2' + \tau_1 \mathcal{H}_{\widehat{\ell}}(\widehat{\mathbf{id}})) \mathbf{b}_4 P_1
\end{aligned}$$

with $s = s' + \tau_2$ which in turn causes the message M_β to be randomised by the quantity $e(P_1, P_2)^{-N_{\tilde{\ell}}\alpha\tau_2 \mathbf{b}_1 \cdot \mathbf{b}_1^*}$ in C_0.

$\mathscr{B}$ returns 1 if $\mathscr{A}$ wins the game and 0 otherwise. Let Y_{real} and Y_{random} denote the events that $\mathscr{B}$ returns 1 in Game_q and Game_{final} respectively. Then,

$$\mathsf{Adv}^{\text{DS1}}_{\mathcal{V}_4}(\mathscr{B}) = |\Pr[Y_{real}] - \Pr[Y_{random}]| = |\Pr[X_q] - \Pr[X_{final}]|$$

□

Theorem E.5. *If the* $(\varepsilon_{\text{DDH1}}, t_{\text{DDH1}})$-DDH1 *and* $(\varepsilon_{\text{DDH2}}, t_{\text{DDH2}})$-DDH2 *assumptions hold in* $\mathcal{G}$*, then* $\mathcal{DPVS}$*-*$\mathcal{HIBE}$ *is* (ε, t, q)-IND-ID-CPA *secure where*

$$\varepsilon \leq \left(1 + \frac{1}{\gamma_p^2}\right)\varepsilon_{\text{DDH1}} + \frac{q}{\gamma_p^2}\varepsilon_{\text{DDH2}} + \frac{q}{p}$$

and $t_{\text{DDH1}} = t + O(h\rho)$, $t_{\text{DDH2}} = t + O(qh\rho)$ *where* ρ *is the maximum time for one scalar multiplication in either of* $\mathbb{G}_1$,$\mathbb{G}_2$.

Proof. Using lemmas E.1, E.2, E.3 and E.4, we have for any t-time adversary $\mathscr{A}$ against $\mathcal{DPVS}$*-*$\mathcal{HIBE}$ in the ind-cpa game, there is an algorithm $\mathscr{B}$ such that

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{ind-cpa}}_{\mathcal{DPVS}\text{-}\mathcal{HIBE}}(\mathscr{A}) &= \left|\Pr[X_{actual}] - \frac{1}{2}\right| \\
&= |\Pr[X_{actual}] - \Pr[X_{final}]| \\
&\leq |\Pr[X_{restricted}] - \Pr[X_0]| + |\Pr[X_{restricted}] - \Pr[X_0]| \\
&\quad + \sum_{k=1}^{q} (|\Pr[X_{k-1}] - \Pr[X_k]|) + |\Pr[X_q] - \Pr[X_{final}]| \\
&\leq \frac{1}{\gamma_p^2}\mathsf{Adv}^{\text{DS1}}_{\mathcal{V}_4}(\mathscr{B}) + q\left(\frac{1}{\gamma_p^2}\mathsf{Adv}^{\text{DS2}}_{\mathcal{V}_4}(\mathscr{B})\right) + \mathsf{Adv}^{\text{DS1}}_{\mathcal{V}_4}(\mathscr{B}) \\
&= \left(1 + \frac{1}{\gamma_p^2}\right)\mathsf{Adv}^{\text{DS1}}_{\mathcal{G}}(\mathscr{B}) + \frac{q}{\gamma_p^2}\mathsf{Adv}^{\text{DS2}}_{\mathcal{G}}(\mathscr{B}) + \frac{q}{p}.
\end{aligned}$$

The theorem now follows directly from Lemma A.1 since n and k are constant in the DS1, DS2 problems we consider. □

E.3 Extending to Anonymous HIBE

The scheme $\mathcal{DPVS}$*-*$\mathcal{HIBE}$ -1 can be extended to an anonymous HIBE scheme that allows zero components in the identity vector usign the same techniques used to obtain $\mathcal{DPVS}$*-*$\mathcal{AHIBE}$. We provide a short respresentation of the scheme here and omit the security proof since it is very similar to that of $\mathcal{DPVS}$*-*$\mathcal{AHIBE}$.

Public parameters:

$$\begin{aligned}
\mathcal{PP} &: (\eta_1\mathbf{b}_1P_1, \ldots, \eta_h\mathbf{b}_1P_1, \theta_1\mathbf{b}_2P_1, \ldots, \theta_h\mathbf{b}_2P_1, e(P_1,P_2)^{N_1\alpha\mathbf{b}_1\cdot\mathbf{b}_1^*}, \ldots, e(P_1,P_2)^{N_h\alpha\mathbf{b}_1\cdot\mathbf{b}_1^*}). \\
\mathcal{MSK} &: (\alpha\mathbf{b}_1^*P_2, \theta_1\mathbf{b}_1^*P_2, \ldots, \theta_h\mathbf{b}_1^*P_2, \eta_1\mathbf{b}_2^*P_2, \ldots, \eta_h\mathbf{b}_2^*P_2).
\end{aligned}$$

Ciphertext.: $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$, $s \xleftarrow{\mathrm{U}} \mathbb{Z}_p$.

$$C_0 = M \cdot e(P_1, P_2)^{sN_\ell \alpha \mathbf{b}_1 \cdot \mathbf{b}_1^*}, \quad \vec{C}_1 = s \sum_{j=1}^{\ell} (\mathsf{id}_j \theta_j \mathbf{b}_2 P_1 + \eta_j \mathbf{b}_1 P_1).$$

Key: $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$, $r_1, r_2 \xleftarrow{\mathrm{U}} \mathbb{Z}_p$.

$$\vec{K} = \alpha \mathbf{b}_1^* P_2 + r_1 \sum_{j=1}^{\ell} (\mathsf{id}_j \theta_j \mathbf{b}_1^* P_2 - \eta_j \mathbf{b}_2^* P_2),\ \vec{D}_{1,j} = r_1 \theta_j \mathbf{b}_1^* P_2,\ \vec{D}_{2,j} = r_1 \eta_j \mathbf{b}_2^* P_2 \text{ for } \ell + 1 \leq j \leq h,$$

$$\vec{J} = r_2 \sum_{j=1}^{\ell} (\mathsf{id}_j \theta_j \mathbf{b}_1^* P_2 - \eta_j \mathbf{b}_2^* P_2),\ \vec{E}_{1,j} = r_2 \theta_j \mathbf{b}_1^* P_2,\ \vec{E}_{2,j} = r_2 \eta_j \mathbf{b}_2^* P_2 \text{ for } \ell + 1 \leq j \leq h.$$

Delegation: $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$, $\tilde{r_1}, \tilde{r_2} \xleftarrow{\mathrm{U}} \mathbb{Z}_p$. Secret key for $\mathbf{id} : \mathsf{id}_{\ell+1}$ is given by

$$\vec{K} \leftarrow \vec{K} + \mathsf{id}_{\ell+1} \vec{D}_{1,\ell+1} - \vec{D}_{2,\ell+1} + \tilde{r_1}(\vec{J} + \mathsf{id}_{\ell+1} \vec{E}_{1,\ell+1} - \vec{E}_{2,\ell+1}),$$

$$\vec{D}_{1,j} \leftarrow \vec{D}_{1,j} + \tilde{r_1} E_{1,j},\ \vec{D}_{2,j} \leftarrow \vec{D}_{2,j} + \tilde{r_1} E_{2,j} \text{ for } \ell + 2 \leq j \leq h,$$

$$\vec{J} \leftarrow \tilde{r_2}(\vec{J} + \mathsf{id}_{\ell+1} \vec{E}_{1,\ell+1} - \vec{E}_{2,\ell+1}),$$

$$\vec{E}_{1,j} \leftarrow \tilde{r_2} \vec{E}_{1,j},\ \vec{E}_{2,j} \leftarrow \tilde{r_2} \vec{E}_{2,j} \text{ for } \ell + 2 \leq j \leq h.$$

setting $r_1 = r_1 + \tilde{r_1} r_2$ and $r_2 = \tilde{r_2} r_2$.

F Anonymous HIBE scheme of De Caro *et.al.* - Comments on the Proof

The technique of Boyen-Waters to achieve anonymity has been used by [SKOS09, Duc10] to contruct anonymous HIBE schemes earlier. In the dual system framework, the only attempt was by De Caro *et.al.* [DCIP10] who considered extending the Lewko-Waters HIBE based on composite order pairing groups to obtain anonymity using the same technique. Keys have two sets of components, the first set for decryption and the second set for rerandomisation during delegation. However, their proof of security of the scheme is incorrect. Here we provide an explanation of why their proof is flawed.

Unlike our strategy, their proof considers a sequence of only $q + 4$ games. This is because they do not use partial semi-functional keys. The keys are changed to semi-functional at once. The problem with their proof surfaces in the second reduction. In order to point out the exact flaw in the argument we provide compact descriptions of the (semi-functional) key and the instance they use in the second reduction. The pairing used is $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ where $|\mathbb{G}| = |\mathbb{G}_T| = N = p_1 p_2 p_3 p_4$ and $\mathbb{G}_m$ denotes the subgroup of $\mathbb{G}$ of order m. Also, we will use multiplicative notation to describe their construction.

Key. Let $\mathbf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell)$ be the identity vector. Choose $R_{i,1}, R_{i,2}, R_{i,j+1}, \ldots, R_{i,\ell} \xleftarrow{\text{U}} \mathbb{G}_{p_3}$ and $r_1, r_2 \xleftarrow{\text{U}} \mathbb{Z}_N$. The key is defined as follows.

$$K_{1,1} = Y_1^{r_1} R_{1,1}\{g_2^{\gamma}\}, K_{1,2} = Y_1^{\alpha} \left(u_1^{\mathsf{id}_1} \cdots u_\ell^{\mathsf{id}_\ell} X_1\right)^{r_1} R_{1,2}\{g_2^{\gamma z_k}\}$$

$$E_{1,j} = u_j^{r_1} R_{1,j}\{g_2^{\gamma z_{1,j}}\} \text{ for } \ell + 1 \leq j \leq h$$

$$K_{2,1} = Y_1^{r_2} R_{2,1}\{g_2^{z\gamma}\}, K_{2,2} = \left(u_1^{\mathsf{id}_1} \cdots u_\ell^{\mathsf{id}_\ell} X_1\right)^{r_2} R_{2,2}\{g_2^{z\gamma z_k}\}$$

$$E_{2,j} = u_j^{r_2} R_{2,j}\{g_2^{z\gamma z_{2,j}}\} \text{ for } \ell + 1 \leq j \leq h$$

Here $u_1, \ldots, u_h, Y_1, X_1 \in \mathbb{G}_{p_1}$ are public parameters, elements $z, \gamma, z_k, (z_{1,j}), (z_{2,j})$ are all indenpendently and uniformly chosen elements of $\mathbb{Z}_N$ and g_2 is generator of $\mathbb{G}_{p_2}$.

Generating key from the instance. In the second reduction it is shown that games $k-1$ and k are indistinguishable. The instance of subspace assumption provided to the simulator is as follows - $(\mathcal{G}, g_1, g_3, g_4, A_1A_2, B_2B_3, T)$ and the simulator has to decide whether $T \in \mathbb{G}_{p_1p_3}$ or $T \in \mathbb{G}_{p_1p_2p_3}$. The parameters are set as: $Y_1 = g_1$, $X_1 = Y_1^b$,$u_i = Y_1^{a_i}$, $1 \leq i \leq h$ where $(a_i), b$ are chosen uniformly at random from $\mathbb{Z}_N$. The k-th key is generated as follows.

$$K_{1,1} = T,\ K_{1,2} = Y_1^{\alpha} T^{z_k} R_{1,2},\ (E_{1,j} = T^{a_j} R_{1,j})_{j=\ell+1}^{h},$$

$$K_{1,1} = T^{r_2'},\ K_{1,2} = Y_1^{\alpha} T^{r_2' z_k} R_{2,2},\ (E_{1,j} = T^{r_2' a_j} R_{2,j})_{j=\ell+1}^{h}.$$

The $\mathbb{G}_{p_3}$ components are generated using g_3.

The flaw in their proof is that certain scalars in the normal and the semi-functional components of the k-th key do not have the proper distribution. As mentioned in the paper, suppose that $T = Y_1^{r_1'} g_2^{s_2} Y_3^{r_3}$ where $s_2 = 0$ or $s_2 \in \mathbb{Z}_{p_2}$. We will only look at the scalars used to compute the $\mathbb{G}_{p_1}$ and $\mathbb{G}_{p_2}$ components.

- The simulator implicitly sets $r_1 = r_1'$ and $r_2 = r_2' r_1'$. These are not independent since r_2' could be zero. But according to construction r_1, r_2 should be independent.
- The proof needs that z_k be information theoretically hidden from the attacker's view. But here a_i's are given out information theoretically and so is z_k.
- The $\mathbb{G}_{p_2}$ components in $E_{1,j}$'s and $E_{2,j}$'s are not independently and uniformly distributed as requried by the definition of semi-functional keys. The exponents of g_2 in $E_{1,j}$'s are scaled by a factor of r_2' to obtain the corresponding exponents in $E_{2,j}$'s.

Intuitively, the root cause of the problem is that keys are changed to semi-functional in one step. Since there are two identity hashes in the key, one instance of the problem does not contain enough randomness to generate both these components. This also justifies our strategy of changing keys in two steps - one for each identity hash.